NASA Conference Publication 10044

Beyond the Baseline

Proceedings of the Space Station Evolution Symposium

Volume 2: Space Station Freedom
Advanced Development Program

Part 1

Proceedings of a conference held at
South Shore Harbour Resort
and Conference Center
League City, Texas
February 6-8, 1990

NASA

Preface

This publication is a compilation of papers presented at the First Annual Space Station Evolution Symposium: Beyond the Baseline on February 6-8, 1990. The symposium focused on the presentation of results by the personnel responsible for advanced system studies and advanced development tasks within the Space Station Freedom Program. The symposium provided an opportunity for dialogue between the users, designers, and advanced planners for Station regarding the long-term utilization of Space Station Freedom.

The papers describe efforts included within the Level I Transition Definition Program to define and incorporate baseline design accommodations which satisfy the requirements associated with potential evolutionary paths, and to develop advanced technology which will enhance Space Station capabilities and enable its evolution. The papers describe work accomplished during fiscal year 1989 and were presented by those in Government, industry, and academia who performed the tasks.

This publication consists of two volumes. Volume 1 contains the results of the advanced system studies with the emphasis on reference evolution configurations, system design requirements and accommodations, and long-range technology projections. Volume 2 reports on advanced development tasks within the Transition Definition Program. Products of these tasks include: engineering fidelity demonstrations and evaluations on Station development testbeds and Shuttle-based flight experiments; detailed requirements and performance specifications which address advanced technology implementation issues; and mature applications and the tools required for the development, implementation, and support of advanced technology within the Space Station Freedom Program.

Dist. 2-21-91

Dr. Earle K. Huckins III
Director, Space Station Freedom Engineering
Office of Space Flight
NASA Headquarters

Listed below are the persons who made this symposium possible.

COMMITTEE MEMBERS

Conference Chair:

- Earle Huckins III
  NASA Headquarters

Program Planning Committee Technical Chairs:

- Stephen Cook
  NASA Headquarters
  Evolution Systems Studies and Analysis
- Gregg Swietek
  NASA Headquarters
  Advanced Development Program

Session Chairs:

- E. Brian Pritchard
  NASA Langley Research Center
- Barry Meredith
  NASA Langley Research Center
- Karen Brender
  NASA Langley Research Center
- Gregg Swietek
  NASA Headquarters
- Paul Neumann
  Space Station Freedom Program Office
- John Muratore
  NASA Lyndon B. Johnson Space Center
- Henry Lum
  Ames Research Center
- Wayne Zimmerman
  NASA Headquarters
- Dale Weathers
  Space Station Freedom Program Office

Administrative Co-Chairs:

- Glenn Freedman
  UH-Clear Lake
- Carla Armstrong
  Barrios Technology

Graphics Coordinator:

- Peter Colangelo
  Omniplan Corporation

CONTENTS – Volume 2, Part 1

Title Page

SESSION VIII: SPACE STATION INFORMATION SYSTEMS
Session Chair: Mr. Robert Nelson, Space Station Freedom Program Office

SESSION IX: ADVANCED AUTOMATION ENVIRONMENTS
Session Chair: Dr. Henry Lum, NASA Ames Research Center

SESSION X: TELEROBOTICS TECHNOLOGY AND APPLICATIONS
Session Chair: Mr. Wayne Zimmerman, Office of Space Station

Time	Topic	Presenter
Monday February 5, 1990		
6:00 p.m. - 9:00 p.m.	Registration	
Tuesday February 6, 1990		
7:30 a.m.	Registration	
8:30	OPENING SESSION	
	Welcoming Remarks	Dr. Aaron Cohen *Director, NASA Johnson Space Center*
9:00	Keynote Address	Dr. William B. Lenoir *Associate Administrator for Space Flight and Acting Associate Administrator for Space Station*
9:30	Space Station Freedom Program Overview	Mr. Richard H. Kohrs *Director, Space Station Freedom*
10:00	Break	
10:30	Human Exploration Mission Planning	Dr. Franklin D. Martin *Associate Administrator for Exploration*
11:00	Mission to Planet Earth	Dr. Shelby G. Tilford *Manager, Geostationary Observations, Mission to Planet Earth*
11:30	Space Station Freedom Evolution	Dr. Earle K. Huckins III *Director, Strategic Plans and Programs Division* *Office of Space Station,* *NASA Headquarters*
12:00 - 1:30	Lunch - Harbour Club	
1:30 - 4:30	SESSION I — EVOLUTION MISSION AND PLANNING Session Chair: Mr. E. Brian Pritchard *NASA Langley Research Center*	
	Long-range Planning for Science Utilization on Space Station	Mr. Robert C. Rhome *NASA Headquarters*
	Space Station Accommodation of Lunar/Mars Exploration Program	Mr. Lewis Peach *NASA Headquarters*
	Advanced Transporation Systems	Mr. Darrell R. Branscome *NASA Headquarters*
	Future European Manned Space Infrastructure	Mr. Jacques Collet *European Space Agency*
	Break	
	Japan's Future Space Activities	Mr. Masatoshi Saito *National Space Development Agency of Japan*
	Canadian Space Activities in the 21st Century	Mr. R. Brian Erb *Canadian Liaison Office*
	Space Station as a Technology Testbed	Dr. Judith Ambrus *NASA Headquarters*
	Life Sciences Planning for Manned Missions	Dr. Arnauld E. Nicogossian *NASA Headquarters*
1:30 - 4:30	SESSION II — TECHNOLOGY AND ADVANCED DEVELOPMENT OVERVIEW Session Chair: Mr. Gregg Swietek *NASA Headquarters*	
	Office of Aeronautics and Space Technology (OAST) Programs	Dr. Judith Ambrus *NASA Headquarters*
	OAST Systems Autonomy and Telerobotics Programs	Dr. Mel Montemerlo *NASA Headquarters*
	Office of Space Flight Advanced Development Activities	Ms. Pat Connor *NASA Headquarters*
	Space Station Freedom (SSF) Flight Telerobotic Servicer	Dr. Harry McCain *NASA Goddard Space Flight Center*
	SSF Advanced Development Program Overview	Mr. Gregg Swietek *NASA Headquarters*
5:30	Reception	
6:30	Banquet with Speaker	

Time	Topic	Presenter
Wednesday February 7, 1990		
8:30 - 12:00	SESSION III — CONFIGURATION EVOLUTION Session Chair: Mr. E. Brian Pritchard *NASA Langley Research Center*	
	Space Station Freedom Integrated Research and Development Growth	Mr. Rudy Saucillo *McDonnell Douglas, Washington, D.C.*
	Space Station Transportation Node Concepts and Analysis	Mr. William Cirillo *NASA Langley Reserach Center*
	Servicing Capability for the Evolutionary Space Station	Mr. Ted Grems *McDonnell Douglas Greenbelt, MD*
	Evolutionary Space Station Fluids Management	Mr. Steve Stevenson *NASA Lewis Research Center*
	Space Station Logistics Systems Evolution	Mr. Michael Tucker *NASA Marshall Space Flight Center*
	Space Transfer Vehicle Accommodations at Transportation Nodes	Mr. Uwe Hueter *NASA Marshall Space Flight Center*
	A Radiological Assessment of Space Nuclear Power Operations Near Space Station	Mr. Steve Stevenson *NASA Lewis Research Center*
	Platform Evolution Studies	Ms. Barbara Walton *NASA Goddard Space Flight Center*
8:30 - 12:00	SESSION IV — FLIGHT SYSTEMS AUTOMATION Session Chair: Mr. Paul Neumann *Space Station Freedom Program Office*	
	Autonomous Power Management and Distribution (PMAD)	Mr. Jim Dolce / Mr. Gale Sundberg / Mr. Jim Kish *NASA Lewis Research Center*
	Laboratory/Habitation Module PMAD Automation	Mr. Bryan Walls *NASA Marshall Space Flight Center*
	Environmental Control and Life Support System (ECLSS)	Mr. Brandon Dewberry *NASA Marshall Space Flight Center*
	PI-in-a-Box	Dr. Larry Young / Dr. Silvano Colombano *Massachussetts Institute of Technology /* *NASA Ames Research Center*
	RCS/RMS Automation using Procedural Reasoning	Mr. H. K. Hiers *NASA Johnson Space Center*
	Thermal Control Expert System	Dr. John Bull / Ms. Kathy Healey / Mr. Jeff Dominick *NASA Ames Research Center /* *NASA Johnson Space Center*
	Summary of Astronauts' Inputs Concerning Automation	Mr. Dave Weeks *NASA Marshall Space Flight Center*
12:00 - 1:00	Lunch - Harbour Club	
1:30 - 4:30	SESSION V — SYSTEM EVOLUTION Session Chair: Mr. Barry D. Meredith *NASA Langley Research Center*	
	Assuring Data Transparency Through Design Methodologies	Mr. Allen Williams *Harris Corporation*
	EVA Systems	Mr. Michael Rouen *NASA Johnson Space Center*
	Data Management Systems	Ms. Katherine Douglas *NASA Johnson Space Center*
	Active Thermal System	Mr. Richard L. Bullock *NASA Johnson Space Center*
	Break	
	Guidance Navigation and Control	Mr. Jerry Kennedy *TRW, Inc.*
	Communications and Tracking	Mr. William Culpepper *NASA Johnson Space Center*
	Structural Analysis of Evolution Station Concepts	Mr. Paul Cooper *NASA Langley Research Center*
	Environmental Control and Life Support System	Mr. Paul Wieland *NASA Marshall Space Flight Center*

Time	Topic	Presenter
Wednesday February 7, 1990 *(continued)*		
1:00 - 4:30	SESSION VI — GROUND OPERATIONS AUTOMATION Session Chair: Mr. John Muratore *NASA Johnson Space Center*	
	Real Time Data Systems for Mission Control	Mr. John Muratore / Mr. Troy Heindel *NASA Johnson Space Center*
	Transition Flight Control Room Automation	Mr. Al Brewer *NASA Johnson Space Center*
	Intelligent Computer-Aided Training Environment	Mr. Bob Savely *NASA Johnson Space Center*
	Platform Management System (PMS) Evolution	Mr. John Hartley / Mr. Mike Tilley *NASA Goddard Space Flight Center*
	Automated PMS Scheduler	Dr. Larry Hull *NASA Goddard Space Flight Center*
	Concepts in Distributed Planning and Control	Dr. Elaine Hansen / Dr. Larry Hull *NASA Goddard Space Flight Center*
Thursday February 8, 1990		
8:30 - 11:00	SESSION VII — OPERATIONS EVOLUTION Session Chair: Ms. Karen Brender *NASA Langley Research Center*	
	Operations Analysis for Evolution Station Concepts	Ms. Karen Brender *NASA Langley Research Center*
	Vehicle Processing Operations Database (VPOD)	Mr. George Ganoe *NASA Langley Research Center*
	On-orbit Assembly and Servicing Task Definition	Mr. Rick Vargo *McDonnell Douglas, Kennedy Space Center*
	Advanced Robotics for In-Space Vehicle Processing	Dr. Jeffrey Smith *NASA Jet Propulsion Laboratory*
	Advanced Automation for In-Space Vehicle Processing	Dr. Michael Sklar *McDonnell Douglas, Kennedy Space Center*
	Space Vehicle Deployment form Space Station	Mr. Paul Henry *NASA Jet Propulsion Laboratory*
	Graphical Analysis of Mars Vehicle Assembly	Mr. Kevin Lewis *NASA Johnson Space Center*
8:30 - 11:00	SESSION VIII — SPACE STATION INFORMATION SYSTEMS Session Chair: Mr. Del Weathers *Space Station Freedom Program Office*	
	Data Management System Advanced Automation	Ms. Katherine Douglas / Mr. Terry Humphrey *NASA Johnson Space Center*
	Operations Management System (OMS) Global Fault Detection / Isolation	Mr. Matt Hanson
	OMS Event Evaluator and Scheduler	Mr. Rick Eckelkamp *NASA Johnson Space Center*
	Automated Software Development Workstation	Mr. Ernie Fridge *NASA Johnson Space Center*
	Technical and Management Information System Design Knowledge Capture	Dr. John Boose / Dr. Jeff Bradshaw / Dr. David Sheema / Dr. Stanley Covington *Boeing Advanced Technology Center*
	Evolution Paths for Advanced Automation	Ms. Kathy Healey *NASA Johnson Space Center*
11:00 - 12:00	Lunch - Harbour Club	
12:00 - 4:00	SESSION IX — ADVANCED AUTOMATION ENVIRONMENTS Session Chair: Dr. Henry Lum *Ames Research Center*	
	CLIPS/Ada Programming Tool	Mr. Chris Culbert *NASA Johnson Space Center*
	ART/Ada Programming Tool	Mr. Chris Culbert *NASA Johnson Space Center*

Time	Topic	Presenter
Thursday February 8, 1990 *(continued)*		
	Knowledge-Based System Verification and Validation	Ms. Sally Johnson *NASA Langley Research Center*
	Intelligent Systems Engineering Methodology	Dr. Bruce Bullock *ISX, Inc.*
	Software Support Environment Design Knowledge Capture	Mr. Tom Dollman *NASA Marshall Space Flight Center*
	Advanced DMS Architectures Testbed	Mr. Terry Grant *NASA Ames Research Center*
	Spaceborne Autonomous Multiprocessor System	Mr. Alan Fernquist *NASA Ames Research Center*
	Information Sciences Experiment System Architecture	Mr. Nick Murray / Mr Steve Katzberg *NASA Langley Research Center*
12:00 - 3:00	SESSION X — TELEROBOTICS TECHNOLOGY AND APPLICATIONS Session Chair: Mr. Wayne Zimmerman *NASA Headquarters*	
	JPL Shared Control Architecture	Dr. Paul Backes / Dr. Samad Hayati *NASA Jet Propulsion Laboratory*
	JPL/KSC Robotic Inspection	Mr. Brian Wilcox / Mr. Leon Davis *NASA Jet Propulsion Laboratory /* *NASA Kennedy Space Center*
	Robotic Assembly of Large Space Structures	Mr. Ralph Will / Mr. Marvin Rhodes *NASA Langley Research Center*
	Space Station IVA Payload Robot	Mr. E. C. Smith *NASA Marshall Space Flight Center*
	Advanced Human-System Interface	Dr. Mike McGreevey *NASA Ames Research Center*

National Aeronautics and
Space Administration

SPACE STATION FREEDOM
ADVANCED DEVELOPMENT PROGRAM
OVERVIEW

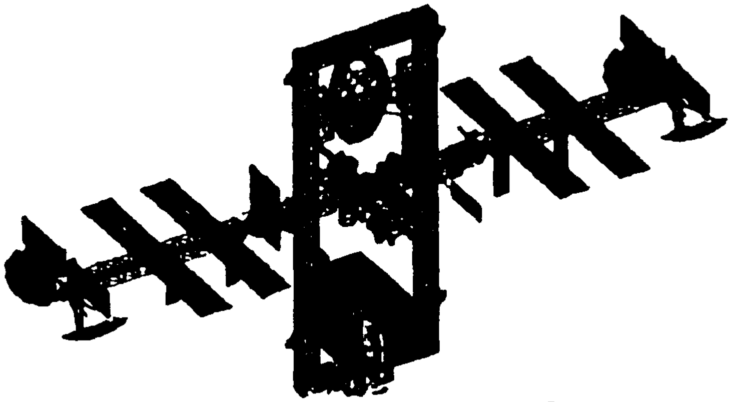
GREGG SWIETEK
Manager
Advanced Development
Office of Space Station

FEBRUARY 6, 1990

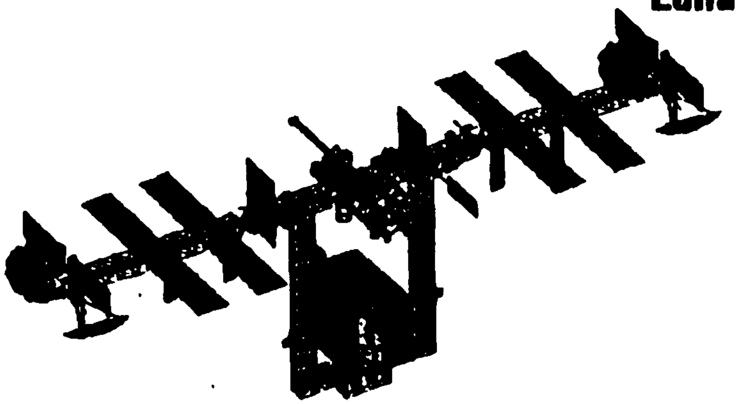
SPACE STATION FREEDOM EVOLUTION

- *Freedom is a permanent facility:*
  - -- Upgrades and configuration changes will take place on-orbit
- *During the operational life of the Space Station:*
  - -- National priorities will change
  - -- User needs and mission requirements will change
  - -- Technology will evolve and components will become obsolete

SPACE STATION FREEDOM – EVOLUTION FOR HUMAN EXPLORATION



Lunar & Mars Operations



Lunar Vehicle Operations

Assembly Complete

OSS-12690
10/19/89

ADVANCED DEVELOPMENT PROGRAM INTRODUCTION AND OVERVIEW

- *Background*
    - -- Transition Definition Program established in FY 1988 to conduct studies and advanced development to ensure baseline Station can evolve
    - -- Modest budget authority placed at Level I as recommended by Phillips Committee
- *Two major elements within Transition Definition:*
    - -- Advanced System Studies
        - Define Station capabilities required to support future needs
    - -- Advanced Development
        - Improve productivity & reduce operations costs of baseline systems

SPACE STATION FREEDOM
ADVANCED DEVELOPMENT PROGRAM

- *Objectives*
  - -- Enhanced baseline Space Station Freedom capabilities
    - Improve productivity & reliability
    - Reduce operations costs
    - Prevent technological obsolescence
  - -- Enable Space Station Freedom evolution
- *Products*
  - -- "Engineering" fidelity demonstrations, evaluations
  - -- Detailed requirements, performance specifications
  - -- Mature technology, tools, applications

ADVANCED DEVELOPMENT PROGRAM PLANNING

- *Advanced Development Task Force held 2/88, LaRC*
  - -- Levels I-III, Code R, Center personnel participated
  - -- Optimizing Productivity selected as theme
  - -- Knowledge-Based Systems (KBS) targeted for FY89 Program

- *Advanced Automation Study completed 5/88*
  - -- Developed short list of baseline candidates
  - -- Analyzed evolution of KBS applications, tools
  - -- Identified high payoff KBS advanced development areas

- *Review of SSFP Advanced Automation Development Capabilities 12/88*
  - -- Provided overview of testbeds with KBS applications
  - -- Described existing/planned operational & support capabilities
  - -- Identified evolution issues and strategies

FACTORS POINTING TO AUTOMATION & ROBOTICS (A&R)

- *Space Station has a 30 year operational life*
  - -- Operations costs, reliability are important concerns
  - -- Incorporation of new technology essential
- *Crew is most scarce resource*
  - -- Productivity is crucial in meeting assembly, user, and servicing requirements
- *Evolution mission scenarios are crew-intensive*
  - -- Science missions will grow & increase demand for crew time
  - -- On-orbit assembly, checkout, launch of Lunar/Mars vehicles

ADVANCED DEVELOPMENT PROGRAM
TASK CATEGORIES - FY 89 - 90

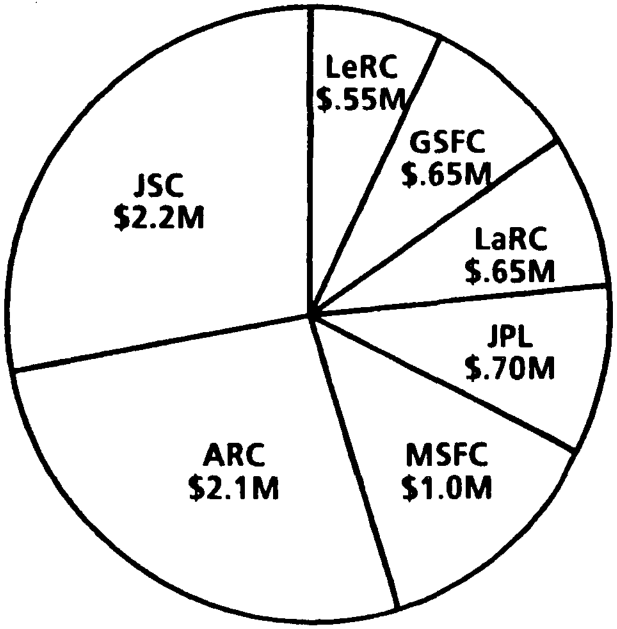
- Flight Systems Automation
- Ground Operations Automation
- Information Processing Systems
- Advanced Automation Software Tools
- Telerobotic Systems

ADVANCED DEVELOPMENT PROGRAM OVERVIEW - FY 89

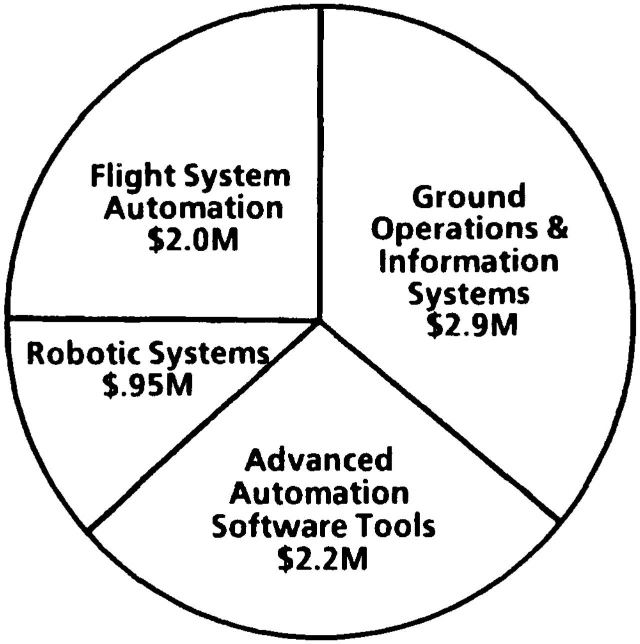
FREEDOM

- 8 tasks funded in FY 88 at $1.8M
- 30 tasks funded in FY89 at $8M

BY CENTER



BY THRUST



- 14 of 30 tasks leveraged; $12.5M from OAST, OSF, USAF, DARPA
  - MOUs/MOA's in place with OAST for Automation & Robotics
  - Draft MOU with OSF for advanced development in final coordination
  - Leverage results in $20.5M impact

FLIGHT SYSTEM AUTOMATION AND GROUND OPERATIONS APPLICATIONS

- *Focused on Automated Status Monitoring, Fault Detection, Isolation, and Recovery (FDIR) using Knowledge-Based System (KBS) techniques*

- *Understand design accommodations ("hooks & scars")*
  - -- Instrumentation, control redundancy, interfaces

- *Identify KBS implementation issues*
  - -- Integration with conventional techniques
  - -- Processing, data storage, communication requirements
  - -- Software development, testing, maintenance
  - -- Boundaries of KBS technology (performance, scale, brittleness)

- *Applications under development for Thermal, Power, Life Support, Data Management, Mission Control*

INFORMATION SYSTEMS
ADVANCED TECHNOLOGY DEVELOPMENT

- *Focused on providing advanced processors, network architectures, and software development tools to prevent obsolescence and reduce power consumption & cost*

- *Enable growth, evolution of OMS, DMS, SSE infrastructure*
  - -- Extensable hardware, software architecture
  - -- Embedded fault tolerance, system security
  - -- Improved processing, memory, data storage, communication network performance
  - -- Reduce training and sustained operations costs

- *Improve software development, testing, maintenance*
  - -- KBS tools for conventional software development
  - -- Tools to develop, test, integrate KBS applications
  - -- Tools for design knowledge acquisition, maintenance

ADVANCED AUTOMATION
SOFTWARE TOOLS

FREEDOM

- Focused on providing programming tools to enable development of integrated KBS applications within the Software Support Environment (SSE)

- KBS programming tools which produce Ada code are under development and evaluation

- "Programmers Assistant" that uses KBS techniques to aid programmer in Ada software re-use under evaluation

- Programming environment for Intelligent Computer-Aided Training (ICAT) applications under development

TELEROBOTICS TECHNOLOGY AND APPLICATIONS DEVELOPMENT

- *Focused on demonstrating technology to improve Flight Telerobotic Servicer (FTS) productivity*

- *Understand integration of component technologies*
    - -- Common architecture for sensing, perception, control

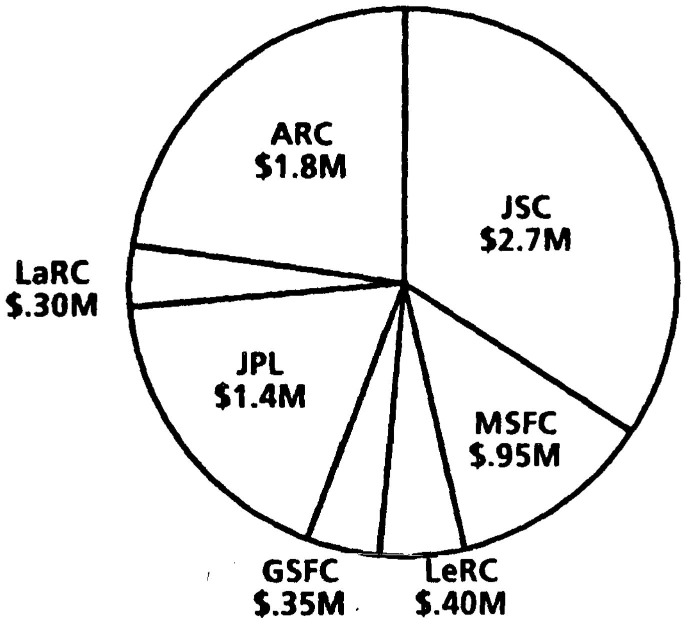
- *Understand integration within total system framework*
    - -- "Robot friendly" design for assembly, maintenance, servicing
    - -- Operational concepts, mission scenarios for EVA/robot mix

- *Understand implications of operations with time delay*
    - -- Impact on interface, control architecture, functionality, task and worksite design

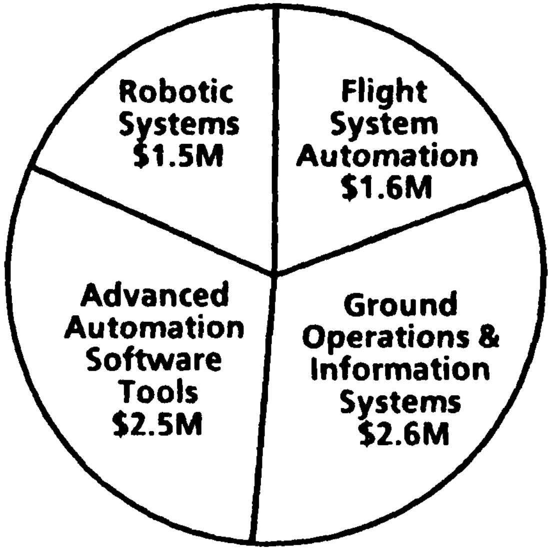
ADVANCED DEVELOPMENT PROGRAM OVERVIEW - FY 90

- 8 tasks funded in FY 88 at $1.8M
- 30 tasks funded in FY89 at $8M
- 35 tasks funded in FY90 at $8.2M

BY CENTER



BY THRUST



- 17 of 35 tasks leveraged; $14.0M from OAST, OSF, USAF, DARPA
  - MOUs/MOA's in place with OAST for Automation & Robotics
  - Leverage results in $22.2M impact

TRANSITION DEFINITON PROGRAM
ADVANCED DEVELOPMENT - FY 1990

- *Flight Systems and Ground Operations Automation Tasks*
  - -- Focused on automated status monitoring, fault detection, isolation, and recovery (FDIR) using knowledge-based system (KBS) techniques
  - -- FDIR KBS applications under development for the Thermal Control System, Power Management and Distribution/Control Systems, Environmental Control & Life Support System, Data Management System, Operations Management System, Mission Control Center (MCC), and the Space Station Control Center (SSCC)
  - -- MCC applications were jointly developed with OAST and OSF and have supported STS-26, STS-29, STS-30, STS-28, STS-34 and STS-32; all will be transitioned to SSCC

TRANSITION DEFINTION PROGRAM
ADVANCED DEVELOPMENT - FY 1990

- *Information Processing Systems*
  - -- Focused on providing advanced processors and network architectures which improve performance and reduce power requirements of Station computer systems; developing advanced Management Information System applications
  - -- Development of a space-qualified multiprocessor is continuing; will be compatible with baseline Station and will be available for on-orbit evaluation by Permanently Manned Capability (PMC)
  - -- Advanced Planning, Scheduling, and Resource Allocations tools which use KBS techniques are under development and evaluation; they enable rapid development of complex plans & schedules with few people, permit accurate real-time re-planning

TRANSITION DEFINTION PROGRAM
ADVANCED DEVELOPMENT - FY 1990

- *Advanced Automation Software*

  -- Focused on providing programming tools to enable development of KBS applications within the Software Support Environment (SSE)

  -- KBS programming tool which produces Ada language software is under evaluation for incorporation into the SSE; KBS "Programmer's Assistant" that aids programmer in re-using existing Ada software is under evaluation; both tools have been transitioned to commercial sector by developing company

  -- Programming environment for Intelligent Computer-Aided Training (ICAT) applications is nearing completion; Initial Station applications will support SSCC console operator training, later use by astronauts for on-orbit refresher training is expected; Private sector funding developed and transitioned high school physics tutor

TRANSITION DEFINTION PROGRAM
ADVANCED DEVELOPMENT - FY 1990

FREEDOM

- *Telerobotic Systems*
  - Focused on improving productivity of Flight Telerobotic Servicer (FTS) and understanding Astronaut-Robot task allocation and robot-system interface issues for evolution
  - Computer-based analysis tool is under evaluation which assesses the allocation of tasks between astronauts and robotic systems; Tool will be distributed to Level II, FTS Project Office, and WP-02 for evaluation and use within the baseline Program
  - Advanced control algorithms and task/spatial planning has been developed and demonstrated and favorably reviewed by Astronaut Office, FTS Project Office and Contractor; Transition to FTS Project Office in support of early demonstration flights and the baseline FTS is underway
  - Development of operator interface, sensor fusion, and control techniques to permit teleoperation and supervised control with time delay will continue; Goal is to support ground operation of FTS while performing inspection, maintenance, & servicing tasks

SUMMARY

- *Advanced Development Program initiated at beginning of Phase C/D with modest funding*
    - High degree of leverage provided by OAST, OSF, other government agencies joint funding
    - Focus on automation and robotics has produced measurable results and provided early options
- *Increased funding in FY91 will expand scope, pace of tasks in preparation for post-PMC sustained operations & evolution*
    - Additional disciplines will be included
- *"Technology transfer is a body contact sport." - John Muratore*
    - People are key factor in affecting or preventing technology migration and utilization

SESSION IV
FLIGHT SYSTEMS AUTOMATION

Session Chair:
Mr. Paul Neumann
Space Station Freedom Program Office

DEVELOPMENT STATUS
AUTOMATION ADVANCED DEVELOPMENT
SPACE STATION FREEDOM ELECTRIC POWER SYSTEM

Feb. 7, 1990

JAMES L. DOLCE
JAMES A. KISH
PAMELA A. MELLOR
LEWIS RESEARCH CENTER, NASA

ELECTRIC POWER SYSTEM AUTOMATION

The paramount objective for our power system's operation is: to generate and dispatch electric power to the loads while maximizing Space Station Freedom's productivity and without violating any constraints. The initial station operation will use dispatchers aided by human-interactive computational facilities to perform the necessary command and control tasks. These tasks constitute planning and decision-making activities that strive to eliminate unplanned outages. To make quality decisions, the dispatchers must have an acumen sharpened through years of experience. Space Station Freedom will adopt an intensive human command and control approach initially, but we perceive that in the long run there are opportunities to reduce our reliance upon skilled dispatchers and to make faster and more consistent on-line decisions by capturing this knowledge in expert systems. The use of such expert systems is shown in this figure. The gist is to perform a closed-loop command and control function using specialized expert systems to perform diagnosis, security analysis, and overall coordination; and to use conventional algorithms for power scheduling and command generation. To develop and demonstrate our automation design we will use the Lewis Space Station Freedom Electric Power Test-bed.

The command and control cycle begins with a sample of data from the test-bed and from the Operation Management System (OMS). Test-bed data is processed by expert systems that recognize and classify the operating state of the power system and then proceed to perform specialized tasks based upon the results of the classification cycle. Operations Management System requests need no special classification software presently.

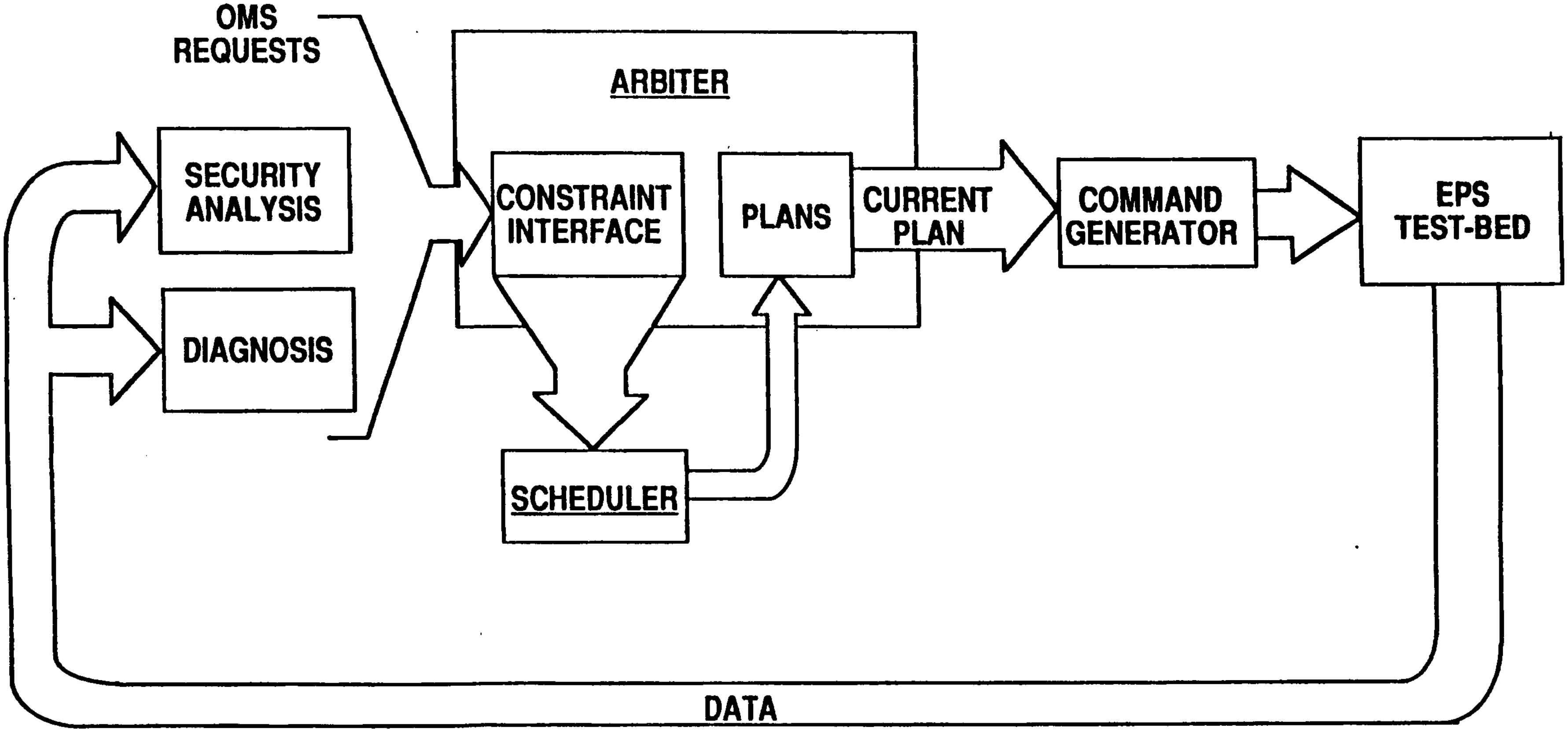
The security analysis software assesses the overload risk from possible failure modes that have been identified beforehand. The system is judged secure if there are no contingencies that result in an emergency situation. Aboard our spacecraft, sudden loss of a power converter is an ever present contingency that produces an emergency state. Converter loss will always produce insecure operation and cannot be alleviated without shedding load. Insecure transmission outages, however, may be prevented by reassigning loads to other busses. These insecure operating conditions are translated into constraints upon the scheduling and distribution of power in the system. For source outages, contingency plans for load shedding must be produced; and for transmission outages, rerouting plans must be produced. The plan formulation and selection is performed by specialized software in the Arbiter expert system.

The diagnosis software determines the most likely cause of abnormal operation. Like the security analysis software it generates constraints upon the scheduling and distribution of electric power.

The Arbiter expert system software coordinates the Operations Management System requests, security analysis results, and failure cause diagnosis by specifying appropriate system operating constraints and electrical loads to a scheduling algorithm. The Arbiter software also determines which schedule and operating plan is to be used given the current state of the power system's operation. This current plan is sent to command generation software which provides the interface between the Arbiter expert system and the computers used to operate the test-bed.

The Scheduler software finds power profiles that maximize productivity and that satisfy the operating constraints stipulated by the Arbiter expert system. The resulting power dispatching schedules repose in memory awaiting selection by the arbitration software.

ELECTRIC POWER SYSTEM AUTOMATION



OMS REQUESTS

SECURITY ANALYSIS

DIAGNOSIS

ARBITER

CONSTRAINT INTERFACE

PLANS

SCHEDULER

CURRENT PLAN

COMMAND GENERATOR

EPS TEST-BED

DATA

AUTOMATION SOFTWARE FOR DIAGNOSIS

Two expert systems are being developed for the diagnosis function:

The first, APEX, has been developed in KEE for use with 20kHZ switchgear. It is a rule-based expert system that uses antecedent driven logic for generating the failure hypotheses and consequent driven logic for deducing the most likely hypothesis. An explanation facility is used to justify the failure cause analysis. The APEX software can accommodate both static and temporal data. The temporal data is used to identify incipient failures. The incipient detection is based on linear regression and correlation analysis. This algorithm finds "soft" failures by detecting graceful degradation in system performance. Rules are used to isolate the cause of the degradation. The addition of temporal data and detection produces an expert system capable of detecting anomalies such as: insulation breakdown in transformers, contact depletion in mechanical switches, and thermal conductivity degradation in power semiconductors.

The second, TROUBLE III, is being developed in ART for use with the photovoltaic generation and nickel-hydrogen battery storage systems. It is an expert system system that uses set-covering rather than a series of if-then rules to encode the failure knowledge. In this software, a data base linking all known system failures to their known symptoms is built and searched to generate the failure cause hypotheses for observed symptoms. Rules control hypothesis generation and determine the most likely cause. The failure knowledge, however, is stored as data and in easily maintained. TROUBLE III uses a standard reliability analysis tool -- the failure modes and effects analysis -- to produce the symptom and failure data base. Symptoms are detected using rule-based classifiers which process static system measurements.

AUTOMATION SOFTWARE FOR DIAGNOSIS

APEX	TROUBLE III
● FOR SWITCHGEAR	● FOR GENERATION & STORAGE SYSTEMS
● RULE-BASED	● SET-COVERING
● KEE	● ART
● DETECTION STATIC TEMPORAL	● DETECTION - - STATIC ONLY

AUTOMATED SOFTWARE FOR SCHEDULING

Four algorithms are being investigated to perform the power scheduling functions: three use integer or mixed integer-linear programming and one uses a value-driven algorithm.

The integer programming approach uses the WS Formulation (after its designers Washington and Sheskin) to represent preferences for load time profiles and their starting times. All variables are integers and the decision variables (when to start a load) are limited to values of 0 or 1. A cardinal value system for starting time preferences is maximized subject to operating constraints using an implicit enumeration technique encoded in a program named ZERON.

The mixed integer-linear programs use either the Washington or the DiFilippo formulations to represent loads, the load's usefulness, and the load's starting times. Both formulations use constraint equations with slack variables to apportion available energy among battery storage and loads, and use 0-1 variables to represent the choices. Both formulations use customized versions of a branch and bound search algorithm to maximize a productivity index.

The value-driven resource allocation program uses a free-market economy model in which consumers (loads) bid for available resources and in which trade-offs among conflicting supply alternatives are governed by cardinal measures of value. Unlike the integer and mixed integer-linear programming methods, the value-driven paradigm schedules not only electric power but also all of the other resource providing subsystem aboard Space Station Freedom. The conceptual design of the allocation algorithm is complete and a report is available. A proof of concept simulation is in progress.

AUTOMATION SOFTWARE FOR SCHEDULING

- **INTEGER PROGRAMMING**
  - WS FORMULATION
  - ZERON SOLVER
- **MIXED INTEGER - LINEAR PROGRAMMING**
  - WASHINGTON FORMULATION
  - DiFILIPPO FORMULATION
  - BRANCH AND BOUND SOLVERS
- **VALUE - DRIVEN RESOURCE ALLOCATION**
  - DECISION-SCIENCE APPLICATIONS, INC. FORMULATION
  - GENERALIZED LAGRANGE MULTIPLIER SOLVER

AUTOMATION SOFTWARE FOR CONSTRAINT INTERFACES

Constraint interface software in the Arbiter expert system has been developed to convert outputs from constraint generating software into properly formulated scheduling problems. These constraint interfaces are for coordinating the APEX diagnostic expert system with the DiFilippo Formulation scheduler and for coordinating the OMS Request data base with the WS Formulation scheduler. Additional constraint interfaces will be developed as the automation design matures.

AUTOMATION SOFTWARE FOR CONSTRAINT INTERFACES

- BETWEEN APEX AND DiFILIPPO SCHEDULER
- BETWEEN OMS REQUEST DATA BASE AND WS SCHEDULER

SIMULATION TO SUPPORT AUTOMATION DEVELOPMENT

- POWER GENERATION AND STORAGE
- SWITCHGEAR FAILURE MODES
- INTERACTIVE EXPERIMENT

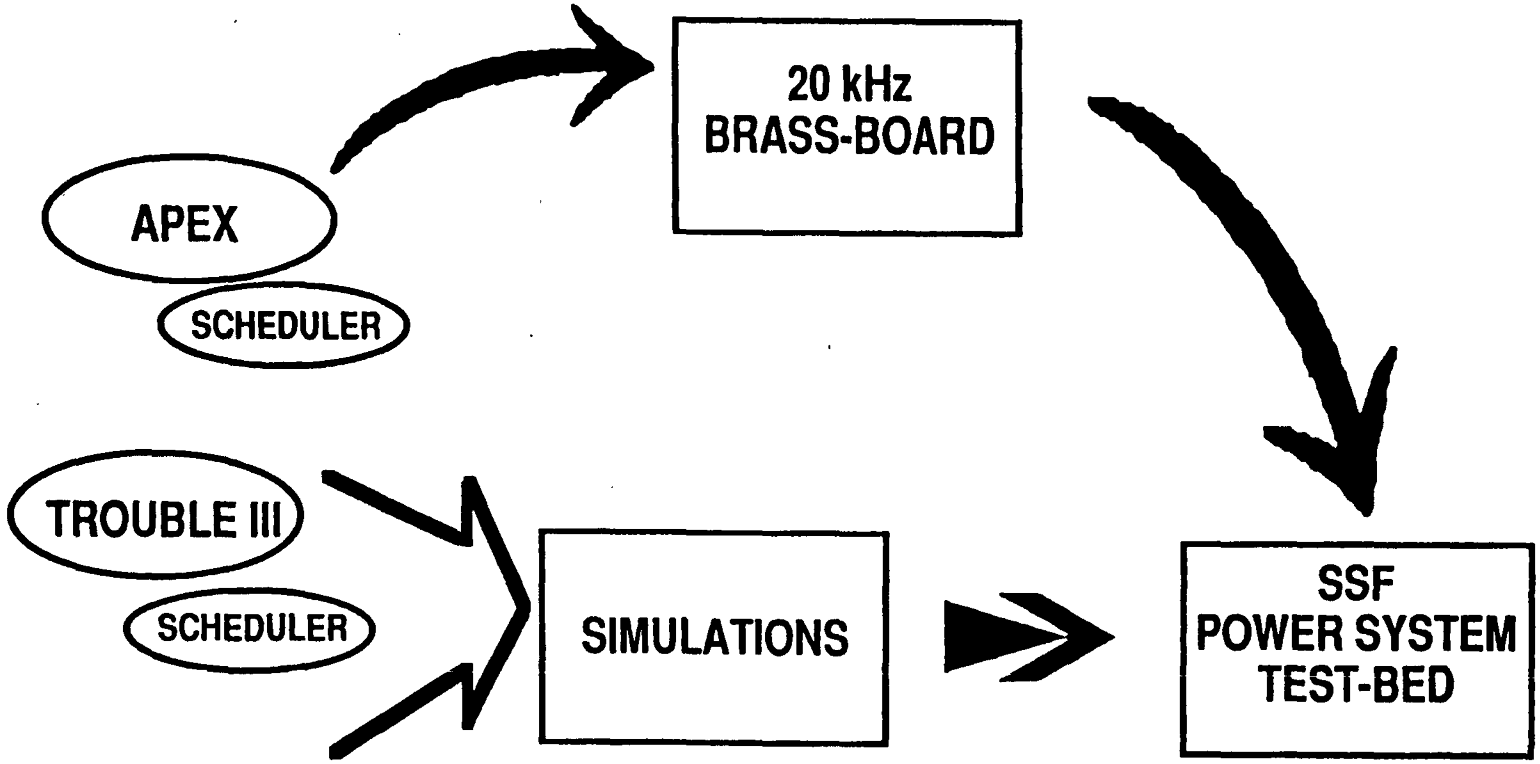
DEVELOPMENT

Two development paths are being pursued: The first uses the APEX switchgear diagnostic system and the DiFilippo Formulation scheduler to produce load shedding or reconfiguration commands for a small 20kHz test-bed. This test-bed (known as the 20kHz Brass-board) contains several pieces of switchgear and a network of microprocessors for gathering data and commanding the switchgear. The thrust of the development is to integrate expert systems with space power hardware, and to learn how expert systems behave in command and control systems. The second uses simulations to provide the behavior of the power system, its computers, payloads, and other station subsystems. The TROUBLE III diagnostics system, the WS Formulation scheduler, and the OMS request data base are being developed with these simulations.

The knowledge gained from the 20kHz Brass-board experiment will be used to guide the automation software development by simulation. The final automation product, a combination of the best performing software, will be evaluated using the Space Station Freedom Power System Test-Bed.

POWER SYSTEM AUTOMATION

DEVELOPMENT



ART/ADA

A development effort is under way to produce Ada versions of the automation software described previously. The objective is to perform a comparative assessment of knowledge-based power system automation developed with LISP-based tools and the same automation developed with an Ada-based implementation of ART. Hardware and software have been procured and personnel from Lewis have been familiarizing themselves with the Ada and ART languages. The first application will be the conversion of the APEX diagnostic system into ART and then into Ada.

ART/ADA

- COMPARE LISP AND ADA IMPLEMENTATIONS
- BEGIN WITH APEX

RESPONSIBILITIES

Funding for the automation development is provided by NASA OAST (Code R) and OSS (Code S). The Code R initiatives focus on technology development for: scheduling, diagnostics, cooperative problem solving using multiple expert systems (e.g. blackboard architectures), and human interfaces to expert systems. All of these technologies are applied to general space power systems with particular emphasis on the Space Station Freedom Power System. The Code S initiatives apply specific technologies to the Space Station Freedom Power System, viz., integration of automation products for diagnosis and resource allocation into the Space Station's power test-bed; and seek to identify the hooks and scars required for successful incorporation aboard the Space Station.

RESPONSIBILITIES

CODE R

- APEX DIAGNOSTIC SYSTEM
- SCHEDULING TECHNOLOGY
- DIAGNOSTIC TECHNOLOGY
- CO-OPERATIVE PROBLEM-SOLVING TECHNOLOGY
- HUMAN INTERFACE TECHNOLOGY

CODE S

- TROUBLE DIAGNOSTIC SYSTEM
- VALUE-DRIVEN RESOURCE MANAGEMENT
- TEST-BED INTEGRATION
- ART-ADA
- HOOKS & SCARS IDENTIFICATION

Space Station Freedom Advanced Development Bimonthly Report for SSM/PMAD Automation

Task Title: **Power Management and Distribution Automation**

WBS Category: Flight Systems

UPN#: 476-81-07

Contract#: Contract NAS8-36433 to Martin Marietta Space Systems

Task Manager:
Bryan Walls
NASA/MSFC, Electrical Power Systems Team/EB12
FTS 824-3311
NASAmail: BWALLS

Alt. Task Manger:
David J. Weeks
NASA/MSFC, Electrical Power Systems Team/EB12
FTS 824-3309
NASAmail: DWEEKS

Significant Events:

During the reporting period of September and October, 1989, several notable milestones were achieved. Foremost, delivery of software for the DC/Star topology change was completed the week of October 16. This delivery allows the highly modified breadboard to be operated by the automation software at about the same level as before the changes in topology and power type occurred. Some obvious bugs were fixed, but the continuing development work is aimed at the mid-1990 delivery on the new workstation and controllers.

All the new hardware has either been acquired or put on order by Martin Marietta. The two Solbourne computers have been delivered to Denver. Five Quimax 80386 computers will be delivered and shipped on to MSFC in the next one or two months, while three more will stay in Denver for development.

A paper describing the intelligent control aspect of the SSM/PMAD breadboard was presented at the IEEE Intelligent Controls Conference by Martin Marietta personnel. Louis Lollar of MSFC also submitted a paper for publication in the Aerospace Applications of Artificial Intelligence Conference, but was not able to attend due to limited travel budget.

Bryan Walls visited Martin Marietta in Denver for a pre-delivery review of progress on the contract. He returned with David Hall, Norma Whitehead, Bob Bechtel, and several MSFC personnel associated with the hardware portion of the contract for an overall strategy meeting for the SSM/PMAD work. These trips were in the last two weeks of September. Seven members of the Martin Marietta team then came to MSFC with the delivery in October.

The communications link between MSFC and LeRC was finally completed in this reporting period. Testing confirmed that computers in the AMPSLAB facility are capable of communicating with machines in LeRC's Power Technology Division. The link uses the TCP/IP protocol over NASA's PSCN-I. LeRC personnel were invited to join MSFC and

Martin Marietta in a meeting to discuss how this resource should best be used to provide cooperation between the two center's PMAD breadboards, but no one was able to come. A proposal was worked up at the meeting, and a copy sent to LeRC, but LeRC has not yet had a chance to concur or disagree with the suggested approach. A copy of what was sent to LeRC is included in the Space Station Freedom Evolution Symposium viewgraphs which are included with the hard copy of this report. It might be best not to include those three viewgraphs in the Annual Report unless it does meet with LeRC approval.

OAST has been undergoing a reexamination of funding commitments, including the funding for cooperating expert systems and intermediate modes of autonomy in SSM/PMAD. Hopefully this will not negatively effect funding levels since a key component of their examination is the level of project support. Since the OAST funding is a major part of funding for SSM/PMAD, it is important that the Space Station Office does show the high level of advocacy that has been demonstrated in the past as OAST asks for input in the coming weeks.

A new MSFC team member has been added to this project. Rajiv Doreswamy will be in charge of the communications effort with LeRC. He has a Masters Degree in Electrical Engineering from Auburn University, and will be a valuable addition to the SSM/PMAD team. The Martin Marietta side of the team is looking for new personnel and are considering hiring a new college graduate.

Lab/Hab Module PMAD

Prepared by
Bryan Walls

for the
Transition Definition
Program Symposium

--Abstract for Transition Definition Program Symposium--
Space Station Module PMAD System

This project consists of several tasks which are unified toward experimentally demonstrating the operation of a highly autonomous, user-supportive power management and distribution system for Space Station Freedom (SSF) hab/lab modules. This goal will be extended to a demonstration of autonomous, cooperative power system operation for the whole Space Station Freedom power system through a joint effort with LeRC, using their Autonomous Power System.

Short term goals for the space station module PMAD include having an operational breadboard reflecting current plans for Space Station Freedom, improving performance of the system communications, and improving the organization and mutability of the AI systems. In the middle term, intermediate levels of autonomy will be added, user interfaces will be modified, and enhanced modeling capabilities will be integrated in the system. Long term goals involve conversion of all software into Ada, vigorous verification and validation efforts, and, finally, seeing an impact of this research on the operation of Space Station Freedom.

Conversion of the system to a DC Star configuration is now in progress, and should be completed by the end of October, 1989. This configuration reflects the latest SSF module architecture. Hardware is now being procured which will improve system communications significantly. The Knowledge Base Management System (KBMS) is initially developed, and the rules from FRAMES have been implemented in the KBMS. Rules in the other two AI systems are also being grouped modularly, making them more tractable, and easier to eventually move into the KBMS.

Adding intermediate levels of autonomy will require development of a planning utility, which will also be built using the KBMS. These changes will require having the user interface for the whole system available from one interface. An Enhanced Model will be developed, which will allow exercise of the system through the interface without requiring all of the power hardware to be operational. The functionality of the AI systems will continue to be advanced, including incipient failure detection.

Ada conversion will begin with the Lowest Level Processor (LLP) code. Then selected pieces of the higher level functionality will be recoded in Ada and, where possible, moved to the LLP level. Validation and verification will be done on the Ada code, and will complete sometime after completion of the Ada conversion.

SSM/PMAD Approach to Automation

- Use fast, simple, dependable hardware at the lowest level as the "first line of defense".
- Provide adequate sensors to understand the system state.
- Distribute processors through the system to control low level hardware, gather sensor data, and communicate with higher level control.
- Coordinate system-wide activity through intelligent central controller(s).

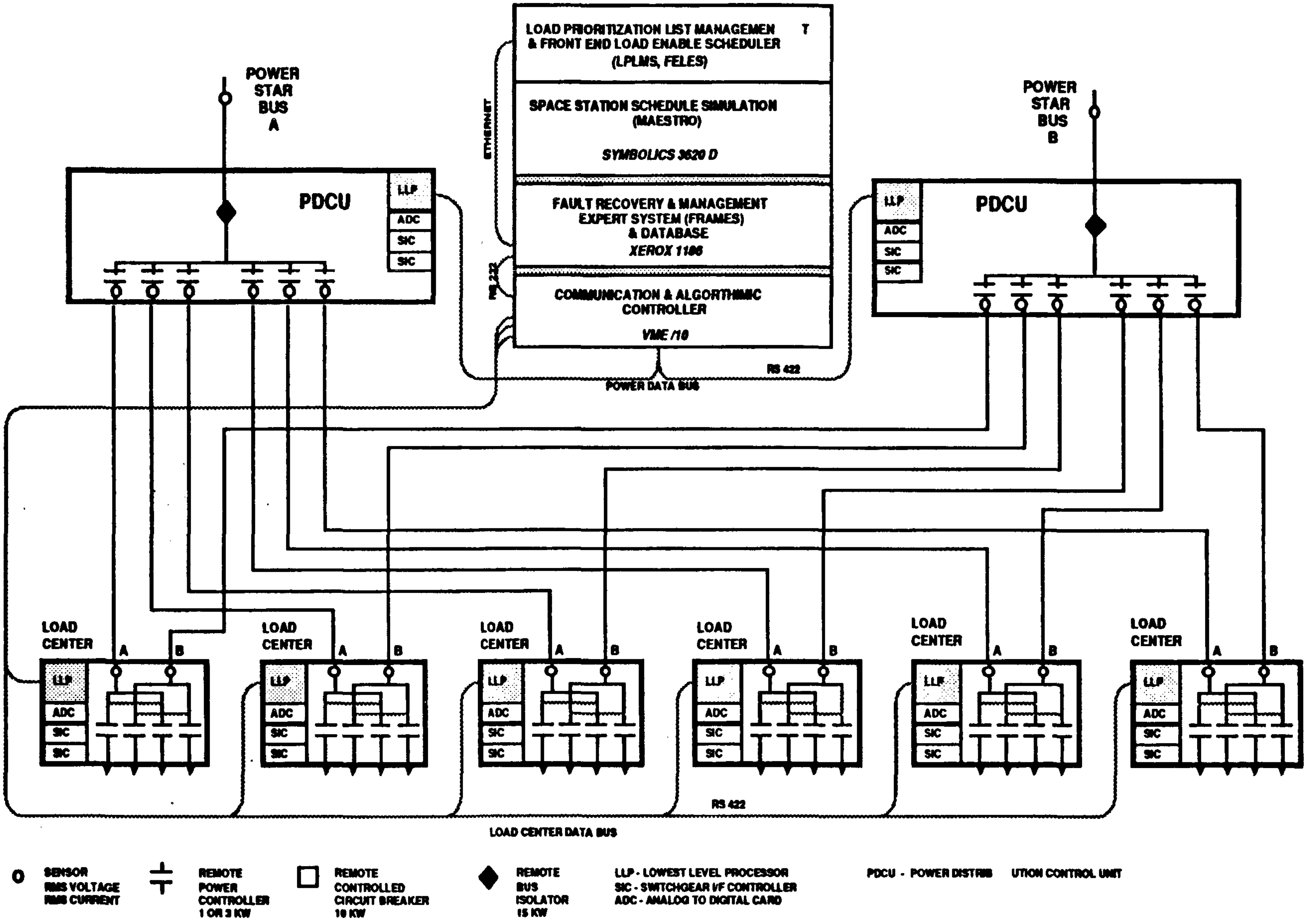
SSM/PMAD Approach to Automation

Circuit breakers are very effective at quickly safing a power system. They represent the only technology available which can react in time and also be remotely controlled and monitored. Though conceivably a very fast computer could read sensors, recognize high current, and order a switch open before damage could occur, it is unlikely with today's technology, less reliable than circuit breakers, and offers no advantage over a remotely switchable circuit breaker.

Knowledge of the actual state of a system is necessary for effective control. Determining sensor locations and designing them into the system, instead of adding them on later, reduces the cost and increasing the reliability of the system as a whole.

A problem with numerous sensors is the flood of data they produce. What does one do with it? The answer proposed here is to sort it out locally, and only pass up summaries unless more is needed. Often "situation nominal" is much more relevant than a stream of data, no matter how accurate.

The central controllers act as an interface for human users, put system in an acceptable state if a problem occurs, assist users in identifying and correcting problems, record and allow modification of the system configuration, and provide the lowest level processors the data they need for normal operation.



SSM/PMAD Topology

Lower Level Autonomy

- Remote Power Controllers (RPCs) provide immediate protection.
- RPCs are grouped into Load Centers. Load Centers are controlled by Lowest Level Processors (LLPs).
- LLPs execute a schedule which is downloaded to them.
- LLPs shed loads which use more than scheduled power.
- LLPs can switch loads to the secondary bus when necessary.
- LLPs communicate with higher level controllers through the CAC.

Lower Level Autonomy -- Switches, Sensors, LLPs, and CAC

The RPCs designed for this breadboard actually consist of two parts: a power stage, which is a switch with resetable over-current protection and a current sensor, and a Generic Card (GC). The GC uses a state machine to offer protection against I^2t, under-voltage, and ground faults, and to communicate with the Switch Interface Card (SIC) and the power stage. Individual sensors are attached to an A-to-D converter which is also attached to the SIC. Each Lowest Level Processor (LLP) communicates with up to two SICs in a Load Center -- one for each bus.

The LLP turns RPCs on or off according to a schedule downloaded to it. It also monitors all the sensors and RPCs. If an RPC trips, the LLP notifies FRAMES of the kind of trip as part of a full status update. The LLP performs in the same way if an RPC is using more power than it is scheduled for, even if the level wouldn't trip the RPC. The LLP orders that RPC off and reports the fault. If the schedule marks an RPC as redundant, the LLP will attempt to turn on a load's redundant RPC if the primary one trips or is shed. Finally, the LLP stores a priority list for it's loads so, in the event of a reduction in system power, lower priority loads will be shed first.

The Communications and Algorithmic Controller (CAC) acts as the communications interface between the LLPs and the higher level controllers. It is the central control point in manual mode operation.

FRAMES

- Monitors breadboard, reporting anomalies to user and to Maestro.
- Evaluates anomalies to determine if failure has occurred, and attempts to diagnose failure.
- Notifies user and other expert systems of conclusions, including any switches considered out of service.
- User interface allows examination of breadboard sensor reading and switch statuses.
- Uses rules developed through work with Power Engineers.
- Coordinate system-wide activity through intelligent central controller(s).

Fault Recovery and Management Expert System
(FRAMES)

FRAMES is one of the three AI systems in the SSM/PMAD breadboard. Each LLP notifies FRAMES any time it recognizes an anomaly, such as tripped breakers or shed loads. Messages giving sensor readings are also sent to FRAMES. FRAMES uses the information which comes to it to characterize the system state. If a failure is diagnosed, it notifies the user via its user interface, and sends a message to Maestro, the system scheduler. Components are marked failed. if it is believed they are broken, or out-of-service if they are not usable (eg. a circuit-breaker above is failed). This information is passed on to Maestro for use in rescheduling.

The FRAMES user interface shows the whole system state. Every switch and sensor in the system is displayed, and shows whether or not it is powered, failed, or out-of-service. Switches also show whether they are opened, closed, or tripped. Components are mousable for further information, including sensor values and values of various flags.

Maestro

- Maestro is a resource scheduler.
- Maestro can schedule numerous activities with multiple constraints.
- Dynamic rescheduling may be done in the event of a fault.
- Maestro will be modified to allow negotiation with the LeRC scheduler.

Maestro

Maestro is a resource scheduler which can schedule numerous activities using multiple constraints. In the SSM/PMAD breadboard the constraints currently used include number of crew members required, equipment resources, and power resources. Power is allocated not just by how much is available to the whole system, but also by the ability of intervening components to supply the power.

Maestro's interface converts the schedule into a list at the component level. Information includes start and stop times and upper and lower power levels at each component.

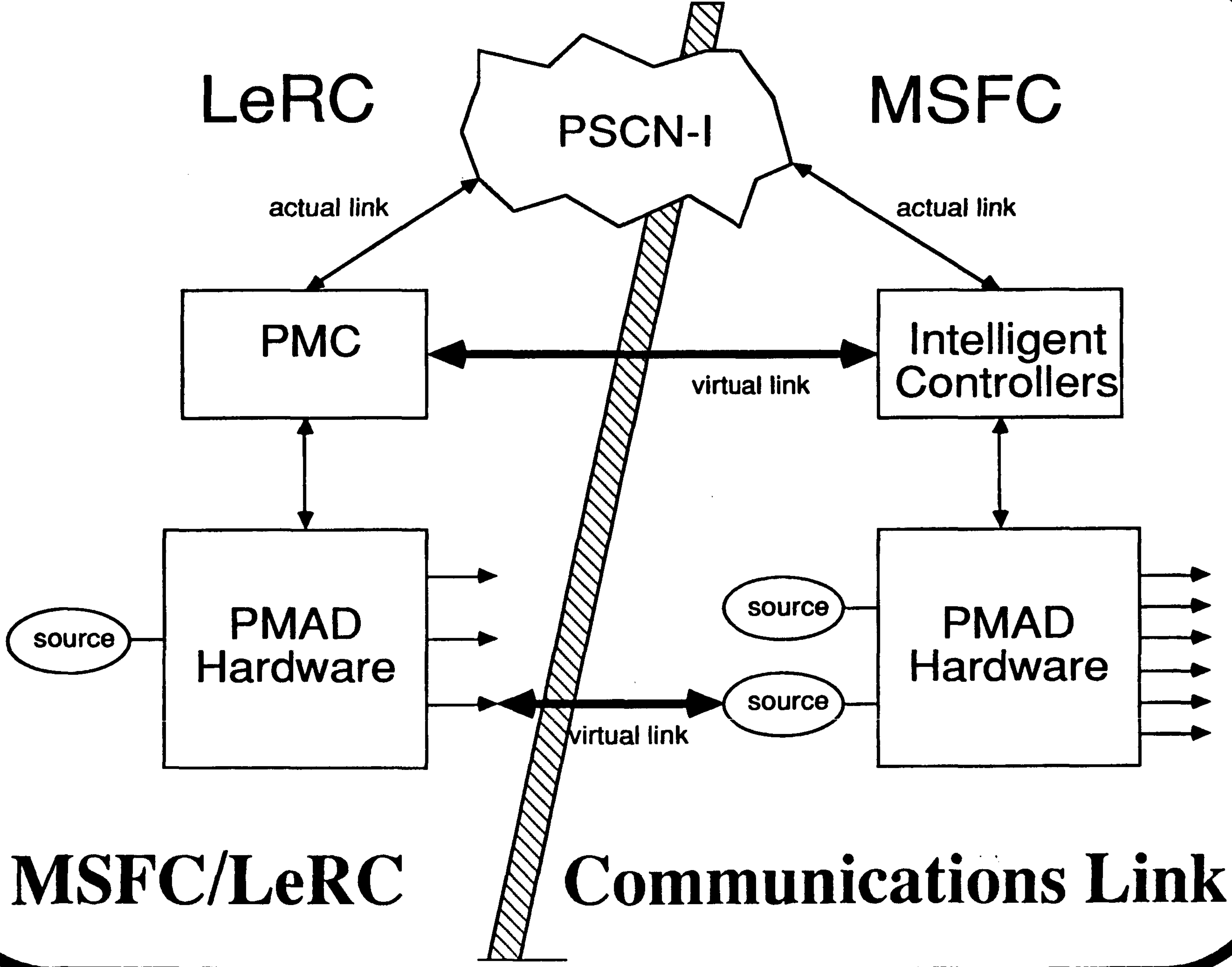
Dynamic rescheduling may be done in the event of a fault. Maestro has access to Activity, Schedule, and Equipment Libraries, and uses encoded knowledge gained from expert schedulers to schedule within constraints.

LPLMS

- The LPLMS periodically notifies the breadboard of the relative priorities of each of the loads.
- This list is updated every 15 minutes.
- The list can be used to prevent high priority loads from being shed in case of a reduction in available power.
- LPLMS can manage dynamic priorities of up to 500 electrical loads.

Load Priority List Management System (LPLMS)

The third of the AI systems, the Load Priority List Maintenance System (LPLMS) uses information from the event list and the activity library, along with its own rules, to dynamically assign relative priority to each active load in the system. A new list is sent down to the LLPs at least every 15 minutes (less than 15 if a contingency occurs). The load priority list can be used to shed loads in case of a reduction in power.



MSFC/LeRC Communications Link

MSFC/LeRC Communications

- A virtual link between the intelligent controllers allows negotiation for power resources.
- A second virtual link between one of the LeRC load and a MSFC source allows emulation of a single breadboard.
- Each breadboard can still be operated independently.
- The actual communications link is available using TCP/IP on the PSCN-I.

Proposed MSFC/LeRC Communications

A communications link is now available between MSFC's AMPSLAB facility, which includes the SSM/PMAD breadboard, and the Lewis Research Center Power Technology Division laboratory, with their Autonomous Power Expert System (APEX). Two virtual links are envisioned between the two PMAD systems.

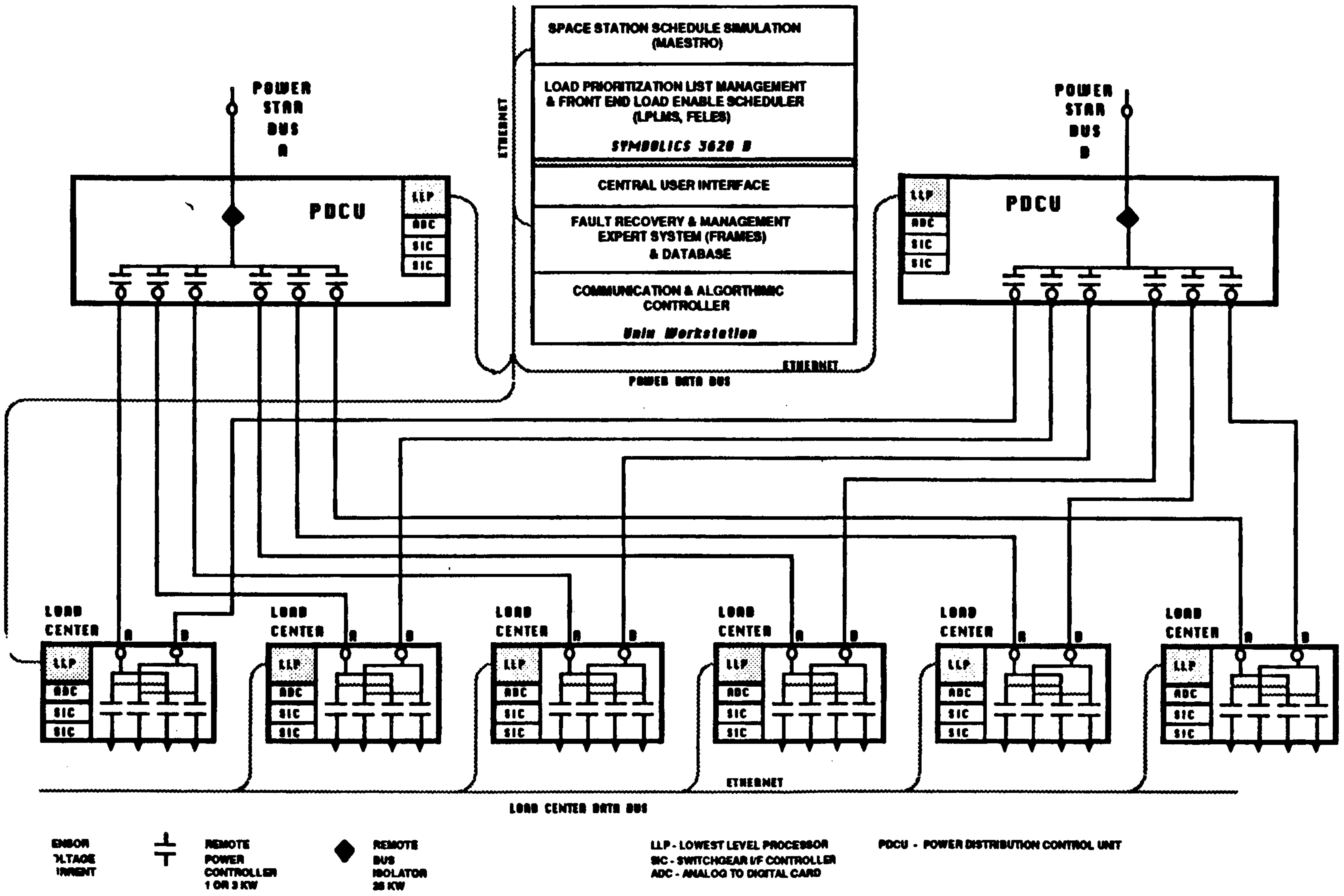
The first link will involve the schedulers for the two systems. Initially interaction will be limited to a request for some level of power from SSM/PMAD for each of the two power buses. APEX would then assign levels, possibly different from those requested, for the buses. As the systems mature, the negotiation will become more sophisticated; SSM/PMAD will provide justification for its request, and APEX will be expected to compare SSM/PMAD's request with those from it's other loads to provide an overall "fair" schedule according to balanced priorities.

The second proposed link will be between one of the loads on the APEX brassboard and one of the dc sources on the SSM/PMAD system. The power drawn by the load will be varied to reflect the power being used in the SSM/PMAD breadboard, thus emulating a single end-to-end power system.

The actual communications link between the centers is via TCP/IP using the PSCN-I service. Both virtual links will be built on this connection, though the second connection may initially be done manually, with communication by telephone.

Planned Changes to SSM/PMAD

- The LLP, CAC, and FRAMES will have new platforms with improved communications.
- FRAMES is being transferred into a new KBMS. Maestro and LPLMS will follow.
- Intermediate levels of autonomy will be added.
- Consolidated, improved user interface will be added.
- Eventual translation to Ada, starting with the lower level autonomy, in two to three years, along with V&V is planned.



Modified Topology

Planned Modifications to SSM/PMAD

By the middle of 1990, some fairly major changes to the automation portion of the SSM/PMAD should be in place. These include a new unix-based computer to host both the FRAMES and CAC functions, 80386 computers with Ethernet to replace the current LLP processors, and some major changes in the structure of FRAMES, with it rehosted in a powerful Knowledge Base Management System environment. The user interface will be significantly upgraded, also.

Under OAST funding, research is under way in how to improve cooperation among the three expert systems, and in adding intermediate modes of autonomy. In the current system, the user has the choice of autonomous operation, or of taking over the whole system. The intermediate modes will provide choices between these two extremes, so a user can have the help of an intelligent assistant.

As the system matures and stabilizes, portions will be transferred into the Ada language, running on general purpose processors. Stricter validation and verification will be observed than is desirable in the present prototypical phase. At the close of this phase, the system should be mature enough to be moved into the mainstream of the Space Station Freedom Program.

Automation of the Environmental Control and Life Support System

Presentation to the Space Station Evolution Symposium, 2/6-8/1990

Brandon S. Dewberry
NASA / MSFC / EB42
MSFC, AL 35812
(205) 544-4247

J. Ray Carnes
AI Center
Boeing Computer Services
Huntsville, AL 35824

Abstract

The objective of the Environmental Control and Life Support System (ECLSS) Advanced Automation Project is to recommend and develop advanced software for the initial and evolutionary Space Station Freedom (SSF) ECLS system which will minimize the crew and ground manpower needed for operations. Another objective includes capturing ECLSS design and development knowledge for future missions.

This report summarizes our results from Phase I, the ECLSS domain analysis phase, which we broke down into three steps: 1) Analyze and document the baselined ECLS system, 2) envision as our goal an evolution to a fully automated regenerative life support system, built upon an augmented baseline, and 3) document the augmentations (hooks and scars) and advanced software systems which we see as necessary in achieving minimal manpower support for ECLSS operations.

In addition, Phase I included development of an advanced software life cycle plan inpreparation for phase II and III, the development and integration phases, respectively. Automated knowledge aquisition, engineering, verification, and testing tools will be used in the development of the software. In this way, we can capture ECLSS development knowledge for future use, develop more robust and complex software, provide feedback to the KBS tool community, and insure proper visibility of our efforts.

Introduction Description

The overall goal of the ECLSS Advanced Automation Project is to help develop a fully autonomous Environmental Control and LifeSupport System for the Space Station Freedom and future manned missions. We have broken this goal into the following more practical objectives:

1) Analyze and document the ECLSS for automation candidates which, when deployed, would minimize crew and ground ECLSS operations.

2) Propose and document a fully automated ECLSS by augmentation of the baselined design with advanced software. Present software hooks and hardware scars which will enable migration of the advanced software to the flight station.

3) Develop, test, and demostrate on ECLSS hardware the most promising automation candidates using tools which maximize productivity in the acquiring, engineering,and storage of ECLSS knowledge.

Our approach is to break the project into phases; analysis, development, and integration:

Phase I/FY89	Analyze and document ECLSS Advanced Automation candidates, approach, and hooks and scars.
Phase II/FY90	Aquire tools and ECLSS knowledge, develop prototype software, and test in a simulated environment.
Phase III/FY91	Integrate the advanced software into the ECLSS advanced development testbed for concrete demonstrations of the advantages of knowledge-based systems diagnosis.

The Johnson Research Center of the University of Alabama in Huntsville has completed the Phase I analysis of the ECLSS.
Boeing Computer Services Artificial Intelligence Center (BCS/AIC) was brought on board late in FY89 as the engineering development contractor in Phases two and three. The AIC has taken part in, and reviewed the UAH work, and developed a detailed software life cycle plan for prototype development and integration.

This presentation summarizes Phase I, gives status of Phase II, and presents a general look at our future plans.

Introduction

- Project Goal:

  A fully automated Environmental Control and Life Support System.for the Evolutionary Freedom Station

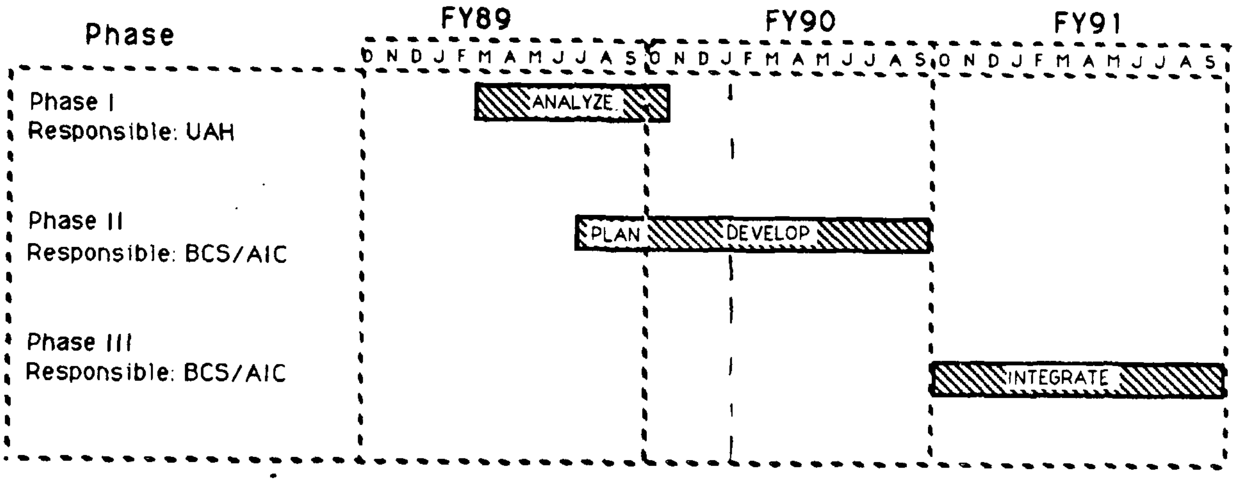
- Practical Objectives:

  1) Analyze and document the baselined ECLSS

  2) Document automated ECLSS: augmentation of baseline

  3) Develop prototype software and integrate with hardware

- Approach



- Presentation Overview
  - Phase I Analysis Approach
  - ECLSS Software Domain Overview
    - Detailed FDIR Description
    - Detailed Water Quality Monitor Description
  - Automation Application Analysis
  - Overview of Major Hooks and Scars Analysis and Results
  - Phase II Development Status and Future Plans
  - Conclusion

Phase I Analysis Approach Description

The Phase I analysis report was generated by UAH in a manner similar to that depicted on the graph. We began by analyzing the ECLSS domain . As the ECLSS is currentlyin the preliminary design stage, our knowledge was generated from three general sources:

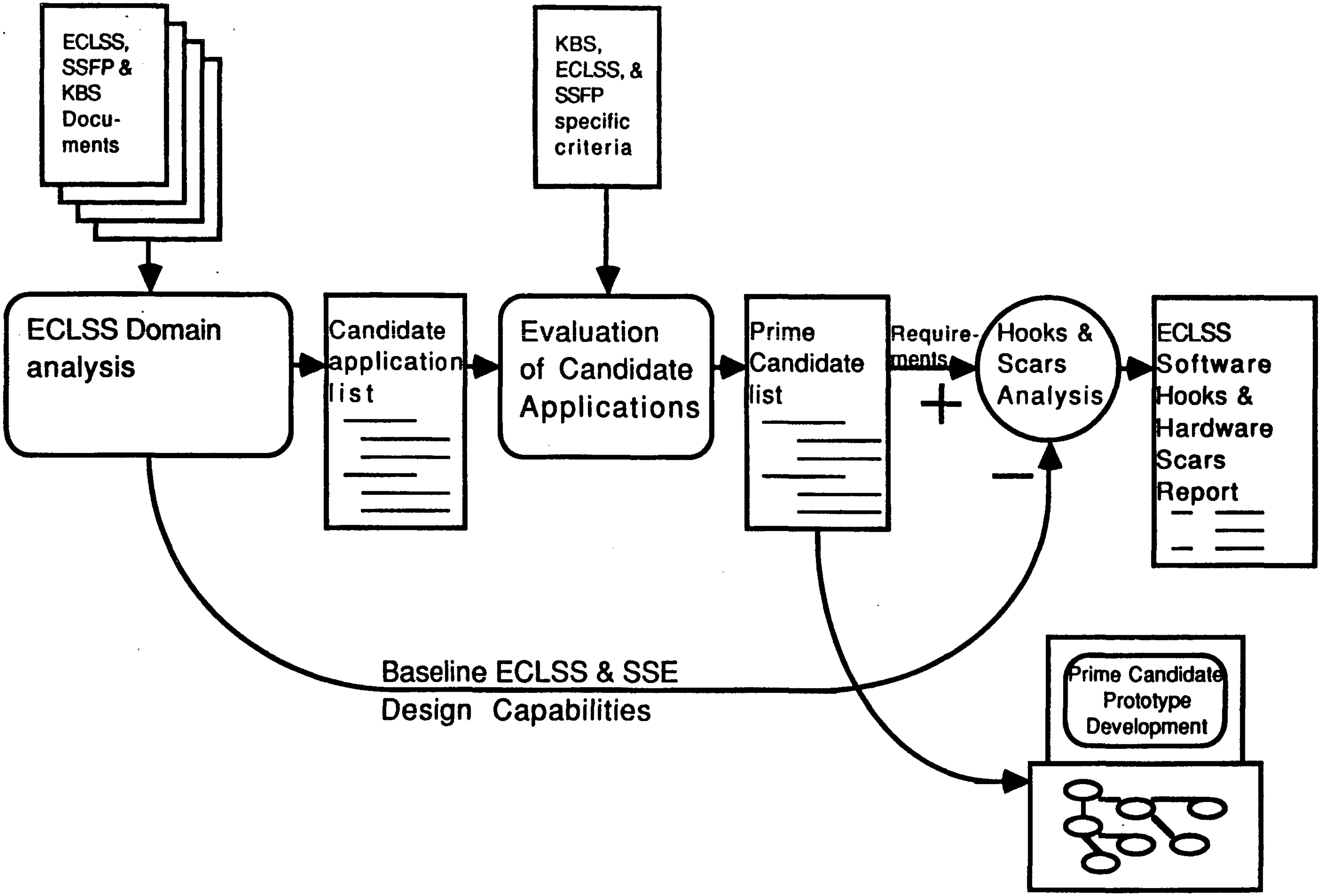
- Applicable Space Station Freedom documentation such as the ECLSS, DMS, OMS, Architecture Control Documents (ACD's), Contract End Item Specifications, ECLSS component test plans, and design review presentations
- Conference reports which discussed control of environmental processes using knowledge-based systems.
- Interviews with ECLSS test and design engineers, scientists, and doctors

The UAH team, consisting of environmental, chemical, process control, and artificial intelligence engineers, gathered some 140 documents and presentations (an appendix to the Phase I report lists these references). They then analyzed each document, determining areas in need of advanced automation and the resulting hooks and scars.

Those software processes which were seen to be candidates for automation (and some new applications, not in the baseline) were listed.
Evaluation criteria was generated and applied to each candidate in order to methodically discuss and document the pros and cons of development of each KBS application.

From the prime candidate list, an application was picked for rapid prototyping, in order todevelop a feel for the resource requirements (speed and memory) and operating system functional interface required. We prototyped a CLIPS based system which monitors and diagnoses faults in the Potable Water Recovery Subsystem. We found we that more than a production system tool was needed for adequate automation of our system (more on this in the results section).

The baselined ECLSS design was compared with the requirements of our candidateadvanced automation systems in order to drive out a list of hooks and scars.



Phase I Analysis Approach

ECLSS Software Domain Overview Description

The ECLSS Station Manager, 1.0) contains four functional software components which: 1.1) Maintain ECLSS configuration data, 1.2) coordinate the ECLSS among elements and, 1.3) control O2N2 pressure. O2N2 Pressure Control, Number 1.3) is an ACS function which is included in the ECLSS station manager software because it must monitor atmosphere constituents throughout the Station.

1.2) coordinating the ECLSS among elements requires expert knowledge of the ECLS System. This function is responsible for Inter-Module Ventilation (IMV), inter-module cabin air, and inter-module potable and hygiene water control. The inter-module air flow problem should be solved during Expanded ECLSS testing when a "race track" of element mockups will be used to make sure no instabilities exist in controlling the blowers and valves which semi-independently push air around the station. The function coordinating the ECLSS among elements function will be defined in greater detail during testing.

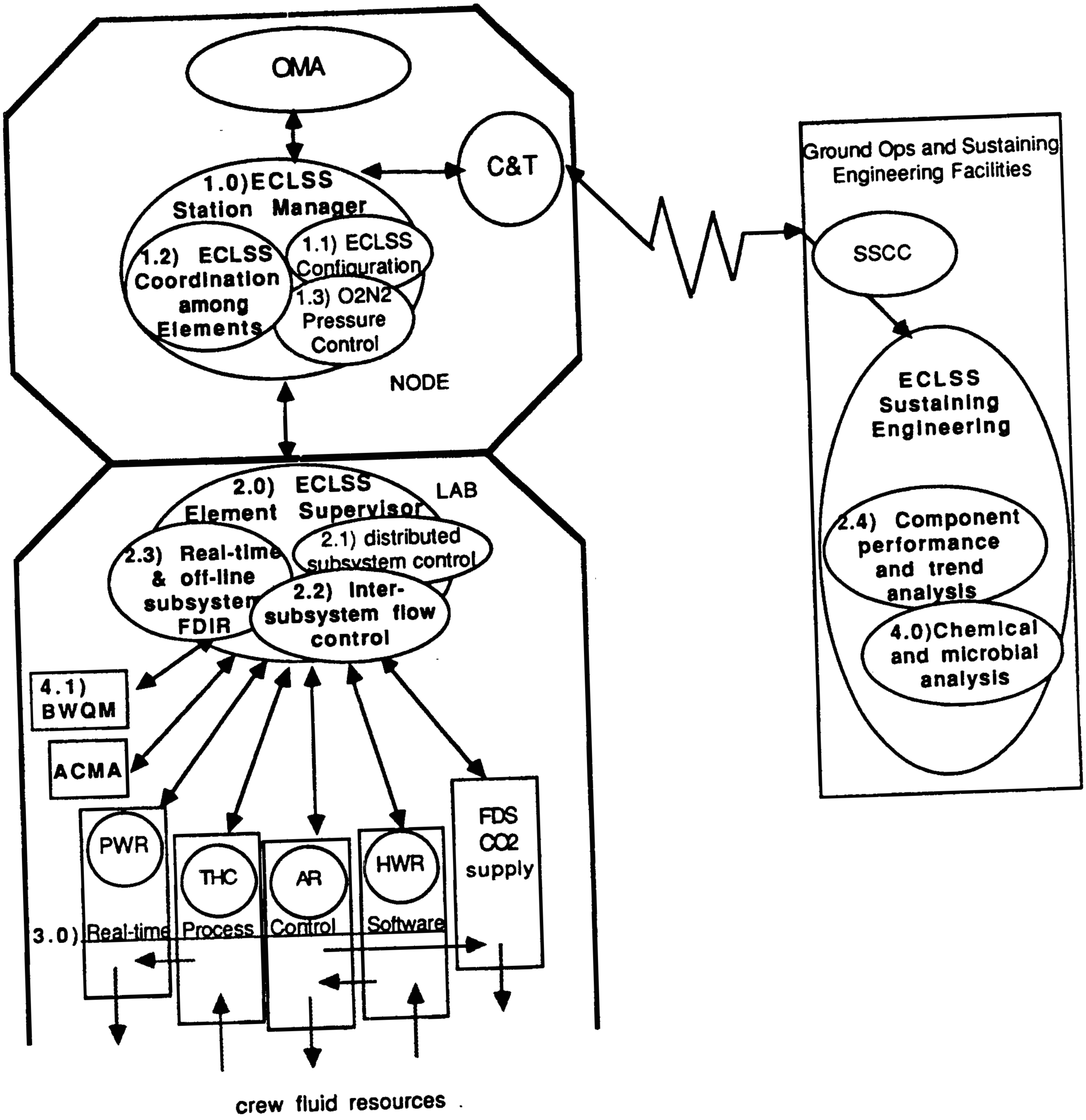
2.0), the ECLSS Element Supervisor contains many candidate automation processes. It includes 2.1), distributed subsystem control, a generic name for those subsystem functions which require distribution throughout the lab, such as Fire Detection sensor monitoring and verification, Avionics air cooling and distribution control, etc.... The process control software loops for these functions will reside in the ECLSS Element Supervisor.

2.2) Inter-subsystem flow control is similar to 1.2) ECLSS coordination among elements in that the total responsibilities of this function will be derived during testing. Its responsibilities will include control of CO2 transfer from the 4BMS to the Bosch, venting from the Bosch to the TCC, water transfer from the Bosch and the THC assembly to the potable water processor's raw water tank, and hygiene water transfer from the hygiene water processor to the water electrolysis unit.

2.3) Off-line subsystem FDIR is a monitoring and diagnosis function. It will be explained later as a prime candidate for an advanced automation approach.

2.4) Component performance and trend analysis is in the ground ECLSS sustaining engineering environment, though some of its functions will migrate on-board as DMS resources permit. This function records and analyzes trend data on the performance of ECLSS pumps, valves, heaters, and filters which will be used to predict faults, maintain system health, and schedule maintenance procedures. Research in chemical and microbial interactions may allow this function to predict and maintain proper chemical and microbial balances throughout the regenerative life support system.

3.0) Real-time Process Control Software consists of process control algorithms in each subassembly, real-time fault detection, and built in tests (BIT) for each subassembly.



ECLSS Software Domain Overview

Real Time and Off Line FDIR Description

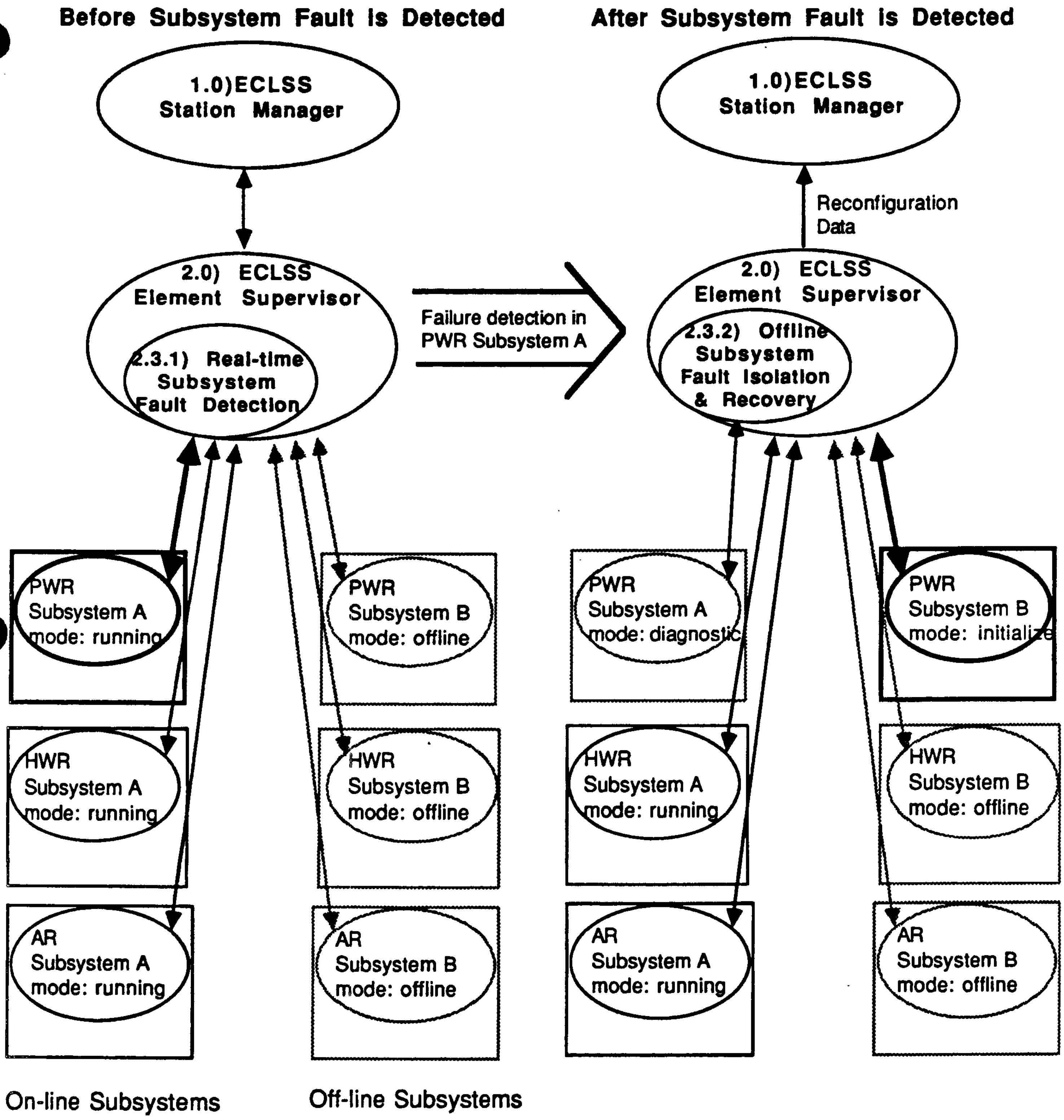
There is a duplication of Potable Water Recovery (PWR), Hygiene Water Recovery (HWR), and Air Revitalization (AR) subsystems for redundancy. There are actually four of these subsystems, two in the Habitation Module, and two in the Laboratory Module. Two of the four are running in nominal operations to support an eight man crew.

2.3) Real-time and Off-line FDIR has been split into its two components, 2.3.1) Real-time Subsystem Fault Detection, and 2.3.2) OfflineSubsystem Fault Isolation & Recovery.

The schenario depicted is a failure in PWR subsystem A. This failure is detected by 2.3.1) Real-time subsystem fault detection, which monitors the sensor values of the running PWR subsystem and compares these to expected values. This system is to be developed as part of the baseline using algorithms implemented in Ada with the support of ground personel. 2.3.1) detects the fault and informs 2.0) the ECLSS Element Supervisor which instructs PWR subsystem B to initialize and PWR subsystem A to change modes to diagnostics. The ECLSS Element Supervisor also starts another software process, 2.3.2) Offline Subsystem Fault Isolation and Recovery, passing it the name of the subsystem to diagnose.

2.3.2), The off-line fault isolation and recovery process inpects the status of the offline (not running but in diagnostic mode) PWR subsystem, sends commands and inspects the responses of the faulty subsystem if the failure is not immediately apparent. The off-line fault isolation and recovery procedure may instruct the faulty PWR subsystem to perform built in tests, or more advanced tests. With the help of ground support and maybe some manual crew procedures, the fault is isolated and a recovery recommendation is formulated and sent to the ECLSS Station Manager through the ECLSS Element manager.

Advancements in the maturity of knowledge based real-time monitoring and diagnosis systems indicate that these software processes are prime candidates for advanced automation development. Autonomous ECLSS subsystem RT & OL FDIR processes could utilize an internal model of the subsystem under test. Implementation of FDIR processes based on internal causual models show strong promise in knowledge based systems for process control diagnosis. Model aqcuisition should start early in the design process, because later implementation may prove too costly.



Real-Time and Offline FDIR

Water Quality Monitor Description

The second major automation candidate is the Water Quality Analysis processwhich includes the process control water quality monitor (PCWQM), not shown on this graph, 4.1) the batch water quality monitor (BWQM), and 4.0) the ground basedchemical and microbial analysis process which is not shown here but is on the ECLSS Domain Functional Schematic.

There are two types of water quality monitors, the PCWQM, and the batch water quality monitor (BWQM).There is a PCWQM associated with each potable, hygiene, and urine pretreatment system. The PCWQM gives near real-time continuous readings for pH, conductivity, iodine concentration, and total organic carbons (TOC) for the product (output) water of each system. If the product water does not match the required specifications for these values, it returns as raw, or input water to the systems.

It is important to understand the limitations of the PCWQM data.
The TOC readings detect the presence of organic compounds but cannot differentiate between compounds or determine their source.
Therefore, periodic manual water sampling and manual analysis will be necessary. The flight and ground batch water quality monitor (BWQM) will provide more complete water quality data.

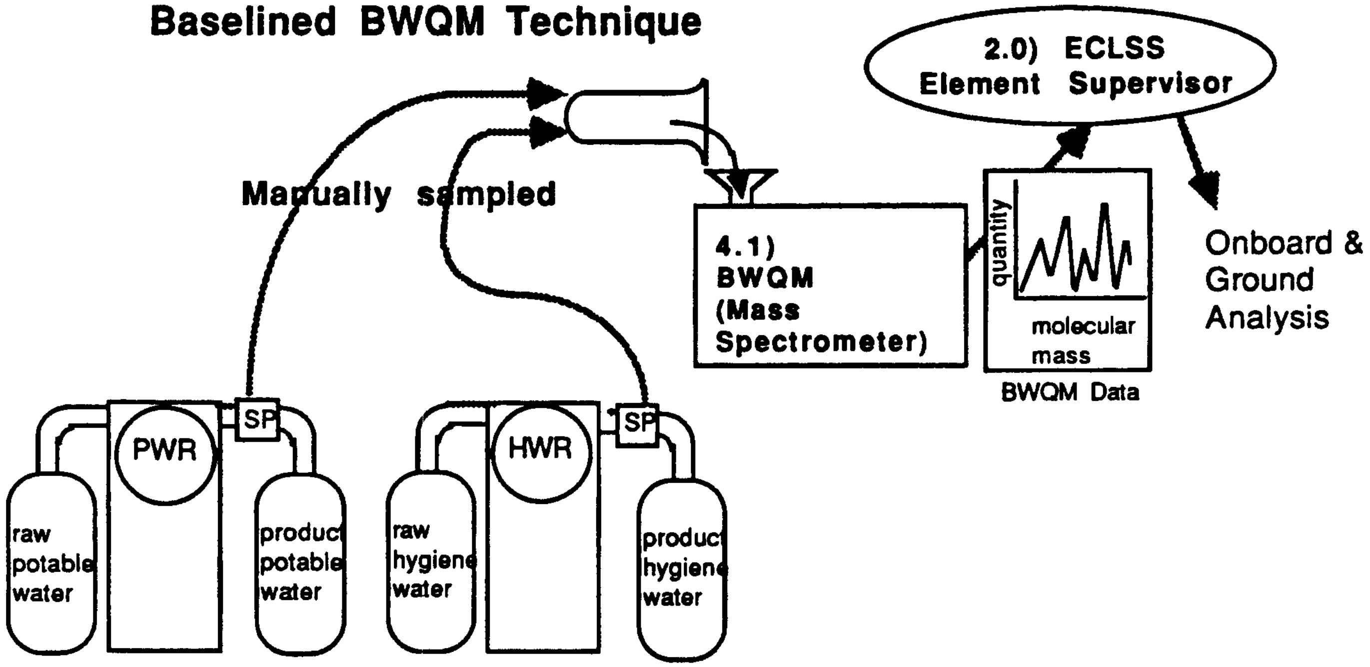
The batch water quality monitor is a mass spectrometer (developed by Perkin Elmer) which requires periodic manual sampling via manual sample ports (SP) in the potable and hygiene product water lines. The on-board BWQM allows the crew to perform tests more frequently than 90 days, when a shuttle flight will return samples for more extensive ground testing. Such testing will include culture growth, a visual inspection, and qualitive judgement by an expert using a microscope to check for various micro-organisms.

Data from the on-board BWQM will be available on the DMS for on-board processing and downlink. Currently, there are no plans to automate the sampling procedures or data analysis.

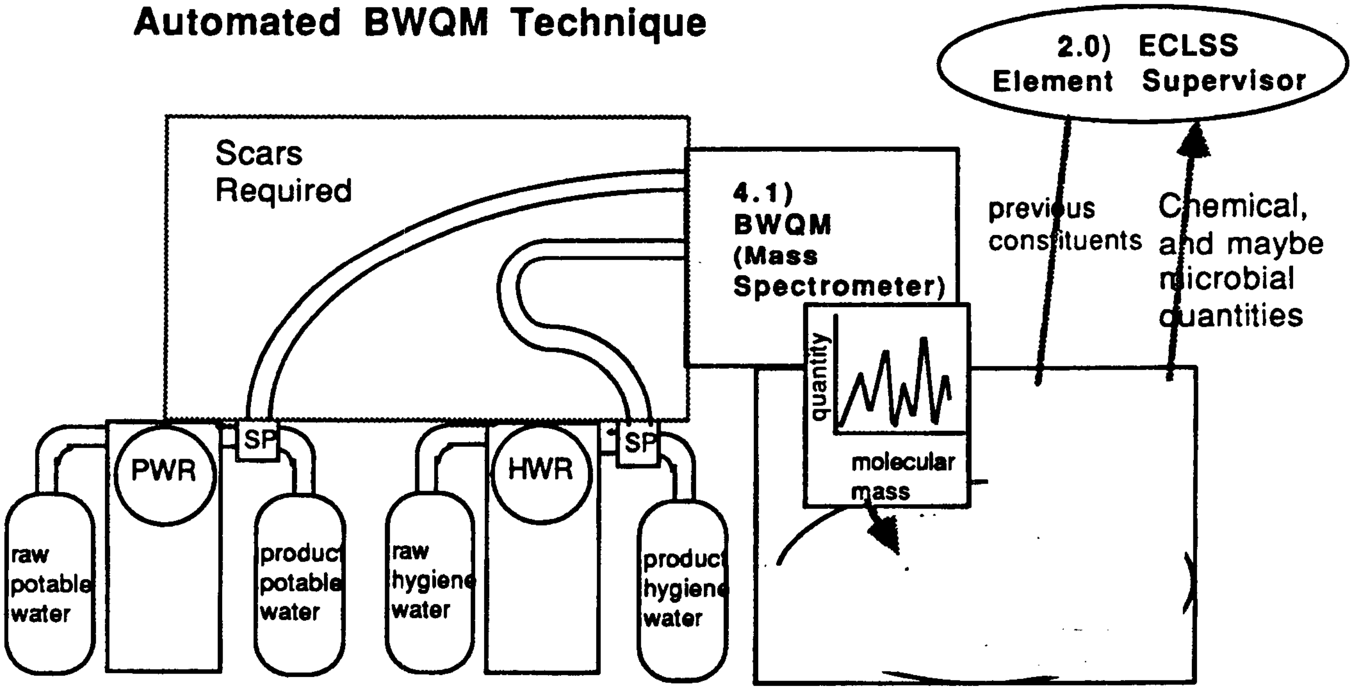
A technique may be needed in the future to automate BWQM sampling of potable and hygiene water. Further, research into automated analysis of mass spectrometer data may enable further automation of this process. This is required to be a real time analysis, on the order of seconds, to feed back into a more advanced control system which will use the data in adjusting a more flexible process control system. This information will also be used to decide if the water was drinkable or okay to wash with.

There has been effort in the medical industry to use many types of analysis including flow cytometry, solid state chemical sensors, and pattern recognition software to isolate specific organic constituents in real-time (the UAH report explores some of this promising research.

Baselined BWQM Technique



Automated BWQM Technique



Automation Application Analysis Description

The objective of this analysis is to determine, based on specific criteria, which function in the ECLSS domain to automate.

1.2) ECLSS Coordination among elements and 2.2) Inter-subsystem flow control. These application are not well understood, expanded ECLSS testing will be required. The baseline application will be accomplished with enhanced instrumentation and traditional algorithmic architectures (Ada tasks and Unix operating system on a 80386-based computer.)

2.3) Real-time and off-line ECLSS subsystem FDIR functions meet all the criteria:

- Implementation of an advanced automation approach to subsystem FDIR will reduce crew and ground maintenance times.
- The processes can be implemented on ground and migrated on board.
- These applications are well understood; the knowledge required is in the designs and models of the subsystems.
- The processes cannot be accomplished with enhanced instrumentation and traditional algorithmic architectures.
- A model oriented approach would minimize the use of sensors for subsystem FDIR and resolve the problem of bad or missing sensor data.
- Technology for advanced automation approach is sufficiently mature due to the emerging capabilities of model based reasoning systems and tools. The subsystem control latencies are sufficiently long to allow implementation of real-time advanced fault analysis.

2.4) Component performance and trend analysis is already in the baseline for ground systems and will be migrated onboard after assembly complete. Some health maintenance functions may require a knowledge based approach. The application is not well understood at this time.

2.5) Automatic and semi-automatic fire suppression is already in the baseline.

4.0) Automatic chemical and microbial water analysis is not well understood. Research is required for this application. Technology does not yet exist for automated extensive analysis of mass spec data, and symbolic and/or neural net processing architectures onboard.

Automation Application Analysis

Criteria For Application Selection

A. Implementation of an advanced automation approach will reduce crew and ground maintenance times.

B. Process can be implemented on ground and migrated on board.

C. Application is well understood.

D. Cannot be accomplished with enhanced instrumentation and/or algoritmic architectures.

E. Technology for advanced automation approachis sufficiently mature.

Candidate Applications	Matching Criteria
1.2) ECLSS Coordination among elements	ABE
2.2) Inter-subsystem flow control	ABE
2.3) Real-time and off-line ECLSS subsystem FDIR	ABCDE
2.4) Component performance and trend analysis	ABDE
2.5) Automatic and semi-automatic fire suppression	CE
4.0) Chemical and microbial water analysis	ABD

Hooks and Scars Analysis Results Overview Description

The following are the major software hooks and hardware scars necessary for evolution to a more autonomous ECLSS. Prototype and baseline ECLSS development will produce more necessary augmentations.

- The advanced subsystem FDIR requires component sensors to be available from the Runtime Object Data Base (RODB) within 1 second with the assumption that the subsystem's control loops have a latency of 5-10 seconds. This allows real-time fault detection and fault preventive reconfiguration to use 3-8 seconds because communication with the ECLSS subassembly monitoring process is expected to take about 2 seconds.

  The software design of the RODB must meet the requirements of process location transparency and performance.

  Our analysis indicates that early capture of design knowledge using design knowledge capture tools such as AQUINAS and object oriented model-based reasoning tools such as KATE and ART/Ada would increase our automation proficiency. We suggest these tools be added to the SSFP Software Support Environment

  ECLSS leak detection can either be implemented using extensive leak detection instrumentation, or by designing subsystems using advanced engineering modeling and design tools which automate determination of optimal placement of leak detection sensors.

  Model-based reasoning approach to subsystem FDIR would allow minimal use of explicit leak detection sensors by inferring leaks using the baseline process control sensors.

- Advanced water quality monitoring will require scarring of the baseline design for future automatic transfer of potable and hygiene product water to the batch water quality monitor. This would be very expensive to implement without the scarring built in to the initial station.

- Research in automated analysis of Water Quality Monitor output is needed.

  Fast processing will be needed in order to implement real time chemical and/or microbial analysis in the life support control system and to quickly determine if the water is drinkable or okay to wash with.

  Intelligent instrumentation systems are needed for real time and inline chemical and microbial analysis.

  Onboard processing may require fast symbolic and/or neural net processing architectures.

- Certain aspects of inter-element coordination and inter-subsystem flow control are candidates for a production system approach. We also explored the use of a blackboard architecture to solve the problems of these functions. They will require expert system support functions in the Application Program Interface Definition (APID), expert system development tools in the SSE, and automated knowledge aquisition systems in the SSE.

Hooks and Scars Analysis Results Overview

- Requirements for advanced subsystem FDIR:

  1) Component sensors available from the Runtime Object Data Base (RODB) within 1 second.
  2) Software process location transparency.
  3) Design knowledge capture tools and object oriented knowledge based system development tools available in the Software Support Environment (SSE).
  4) Engineering modeling and design tools which automate determination of optimal placement of leak detection sensors.

- Requirements for advanced water quality monitoring:

  1) Scarring for automatic transfer of potable and hygiene product water to the batch water quality monitor
  2) Research in automated analysis of water quality monitor output
  3) Research in realtime chemical and microbial analysis
  4) Probable: symbolic and/or neural net processing architectures onboard

- Requirements for advanced ECLSS inter-element coordination

  1) Expanded ECLSS domain testing.
  2) Knowledge based system support functions in the Application Program Interface Definition (APID)
  3) Knowledge based system development tools in the SSE.
  4) Automated knowledge acquisition systems in the SSE.

- Requirements for inter-subsystem flow control:

  1) Expanded ECLSS domain testing.
  2) Blackboard software architecture application to the intersubsystem flow control problem.
  3) Blackboard development tools in the SSE.
  4) Automated knowledge acquisition systems in the SSE.

Development Status and Plans Description

Status

We have used Aquinas for knowledge acquisition on the potable water processor tradeoffs.

ART/Ada has been installed.

Development of the model based Water Recovery and Air Revitalization Diagnosis prototype using KATE is on-going.

Plans

Demonstration of the KATE based Water Recovery Diagnosis Prototype using a simulation of the Water Recovery Subsystem.in the summer of FY90.

Demonstration of the ART/Ada based Potable Water Recovery Diagnosis Prototype was scheduled for 2/90 but will be delayed until this spring. Possible use of TAE+ as the interface.

Phase III will demonstrate the Water Recovery Diagnosis Prototype using actual ECLSS Water Recovery Subsystem hardware.

Also in Phase III we will begin development of the Air Revitalization Diagnosis prototype which will contribute to the overall Regeneration Analysis and Diagnosis system.

A fourth Phase is needed to produce results on expanded ECLSS test data and to complete the Regeneration Analsys and Diagnosis system.

Development Status and Plans

Status

- Knowledge Acquistion
- ART/Ada beta test software installed
- Model based water recovery subsystem diagnosis development using KATE

Plans

Phase II

June/FY90	Demonstration of the Potable Water Recovery Diagnosis Prototype port to ART/Ada
August/FY90	Demonstration of the model based Water Recovery Diagnosis Prototype using a simulation of the Water Recovery Subsystem

Phase III

August/FY91	Demonstration of the model based Water Recovery Diagnosis Prototype using actual ECLSS Water Recovery Subsystem hardware
FY91	Development and integration of Air Revitalization Diagnosis with demonstrations using simulation and actual hardware Preliminary Integration of a Regeneration Analysis and Diagnosis system

Phase IV

FY92	Results on expanded ECLSS test data Completion of the Regeneration Analsys and Diagnosis system.

Conclusion

Phase I results:

- ECLSS Advanced Automation Candidates.
- ECLSS hooks and scars analysis for growth to advanced automation.
- Prototype of the Potable Water Recovery FDIR Knowledge Based System.
- Advanced software life cycle plan for development and integration.

Phase II Status:

- Knowledge Acquisition using Acquinas.
- ART/Ada port of the CLIPS Water Recovery Diagnosis prototype in progress.
- KATE - Based Potable Water Recovery Diagnosis prototype

Phase II Development:

- Demonstrateion using simulation of the Water Recovery Subsystem.
- Continued ART/Ada beta testing.

Phase III Plans:

- Demonstration using Water Recovery Subsystem hardware in testbed.
- Integration of Air Revitalization Diagnosis knowledge

Phase IV Plans:

- Demonstration of Regeneration Analysis and Diagnosis system on expanded ECLSS test bed.

Conclusion Description

The results of Phase I of the ECLSS Advanced Automation project were discussed. These results include:

> Analysis and documentation ofthe ECLSS Advanced Automation Candidates which support our Phase II development, and baseline system design augmentations required for easier growth to automation.
>
> Development of a prototype Potable Water Recovery Diagnosis rule based system which helped in our requirements analysis and will be used as a starting point for future development.

Phase II development status was discussed and included the use of AQUINAS for knowledge acquisition. ART/Ada and KATE for development of the ECLSS Water Recovery and Air Revitalization Diagnosis software.

Phase II will produce a Water Recovery Diagnosis Prototype which will be demonstrated using a simulation of the Water Recovery Subsystem.

Phase III will demonstrate the Water Recovery Diagnosis Prototype using actual ECLSS Water Recovery Subsystem hardware. Also in Phase III we will begin development of the Air Revitalization which will contribute to the overall Regeneration Analysis and Diagnosis system.

A fourth Phase is needed to produce results on expanded ECLSS test data and to complete the Regeneration Analsys and Diagnosis system.

PI-in-a-box

An Expert System to Advise Astronauts During Experiments

PI-in-a-box

An Expert System to Advise Astronauts During Experiments

The project was made possible by NASA grant NCC 2-570 and RTOP 506-47-11 for "Crew Station Design," respectively from the AI Research Branch and the Human Factors Division at NASA-Ames. The Stanford University Knowledge Systems Laboratory and Apple Corporation also provided generous support.

Outline

- The Problem
- The Solution
- System Architecture
- Protocol Manager
- Interesting Data Filter
- Data Analysis and Quality Monitoring
- User Interface Issues
- Explanations
- Conclusions

Outline:

We describe the overall architecture, comprised of several modules, and discuss in greater detail our work on two of these modules which are central to the philosophy of this system: the Protocol Manager and the Interesting Data Filter. Finally, we address some engineering issues related to data acquisition and monitoring and to human factors.

The Problem

Time and resource limitations severely limit flexibility during space experimentation:

- PI is physically distant from experiment.
- Communication is often of insufficient bandwidth or not timely enough.
- Experiments are numerous and varied.
- Space Station environment is likely to aggravate the situation.

The Problem:

Perhaps the scarcest resource for manned flight experiments - on Spacelab or on the Space Station Freedom - will continue to be crew time. To maximize the efficiency of the crew, and to make use of their abilities to work as scientist collaborators as well as equipment operators, normally requires more training in a wide variety of disciplines than is practical.

In a typical laboratory setting the Principal Investigator (PI) is able to exert direct control over all aspects of an experiment, and his or her expertise can be brought to bear as events unfold, to correct problems or to follow new leads. This kind of flexibility is currently lacking during space experimentation, both due to time and resource constraints, and due to the physical distance from the PI at the time of the experiment. Communication is often not sufficient or not timely enough to bridge this physical gap.

Furthermore, astronauts are trained to perform a large number of experiments in different fields, and cannot be expected to acquire the in-depth knowledge required to deal effectively with all unexpected contingencies. This problem will be exacerbated in the Space Station era, with its longer tours and larger number of experiments.

Two Approaches:

- Use Telescience to improve communication between PI and astronaut experimenters.
- Use Artificial Intelligence and Expert Systems technology to put PI's expertise on board.

Two Approaches:

Two approaches are being pursued by the space science community to alleviate these problems. One has been named "telescience" and its aim is to use communications technology to "bring" the space experiment into the PI's lab. A second approach, described here, aims to apply Artificial Intelligence technology to bring the PI's expertise on board.

These two approaches are not mutually exclusive, and we do not intend to debate the merits of each. Here we illustrate several aspects of our ongoing work on a system named "PI-in-a-box", the purpose of which is to bring the PI's expertise close to where experimental events are occurring.

"PI-in-a-box"

An expert system to help astronauts perform in-flight experiments.

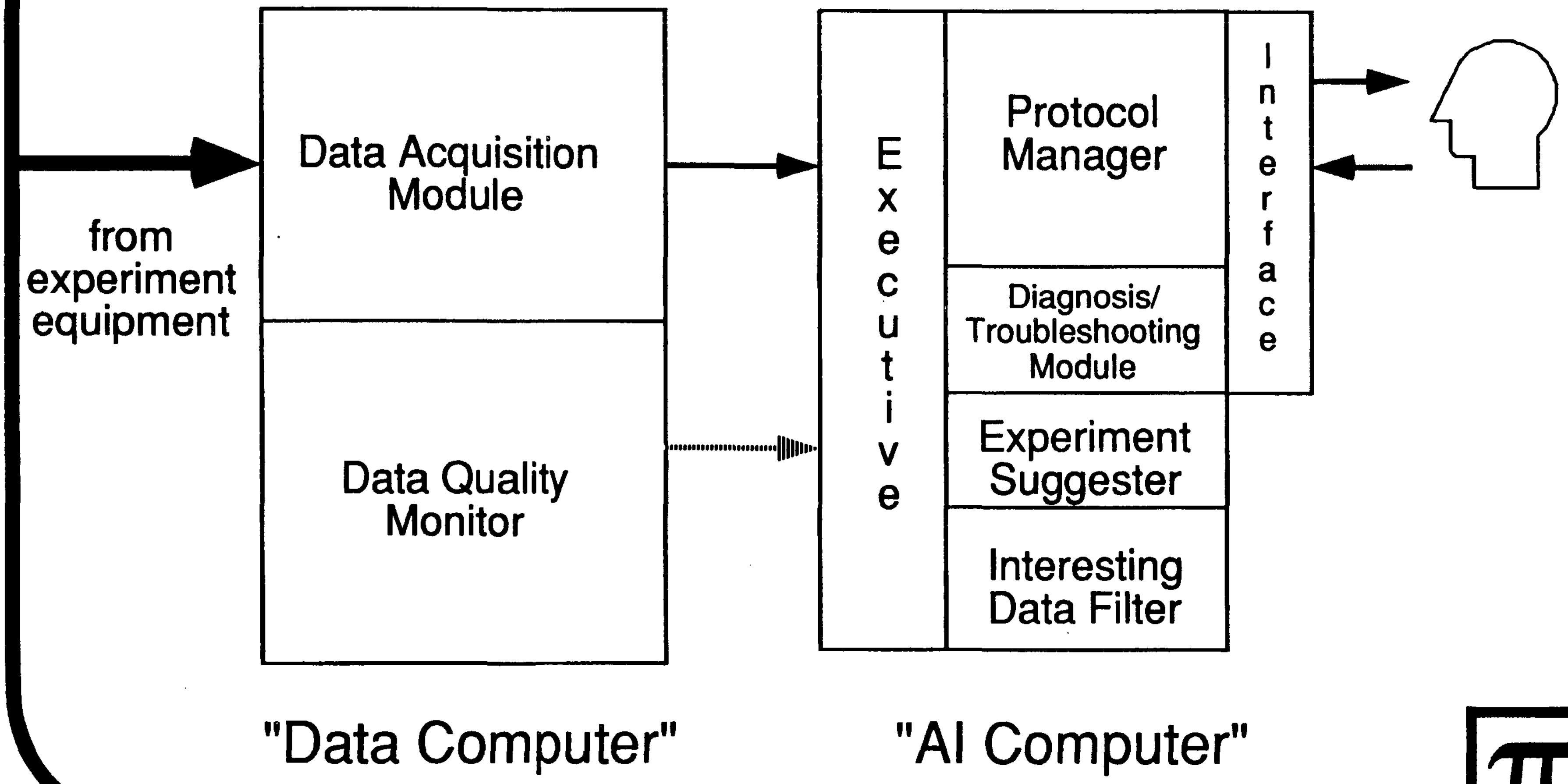
GOALS:

- Identify and permit investigation of "interesting" data.
- Suggest protocol changes that would result in better utilization of remaining time.
- Capture, reduce, and archive experimental data.
- Monitor data quality and help diagnose problems with equipment when experimental data is erratic or poor.

"PI-in-a-box"

The successful application of on-board expert systems, as envisioned by the "Principal Investigator in a Box" program, should alleviate the training bottleneck and provide the astronaut with the guidance and coaching needed to permit him or her to operate an experiment according to the desires and knowledge of the PI, despite changes in conditions. In addition to the functions of providing expert advise concerning scheduling and repair, the program should bring the astronaut into the scientific evaluation phase of an experiment by sharing with him the guidance and observations regarding the relevance and importance of data as it is being generated.

System Architecture



from experiment equipment

Data Acquisition Module

Data Quality Monitor

Executive

Protocol Manager

Diagnosis/ Troubleshooting Module

Experiment Suggester

Interesting Data Filter

Interface

"Data Computer"

"AI Computer"

System Architecture

The PI-in-a-box system is comprised of eight modules:

- The **Data Acquisition Module** (DAM) collects and reduces the raw data from the on-board experiment equipment.
- The **Data Quality Monitor** (DQM) ensures that the incoming data is reliable and error-free.
- The **Protocol Manager** (PM) helps keep the experiment on schedule by monitoring the experiment's progress and suggesting modifications to the protocol when necessary.
- The **Interesting Data Filter** (IDF) recognizes experimental data that is likely to be "interesting" to the PI, and helps the protocol manager suggest ways to pursue the interesting results.
- The **Diagnostic and Troubleshooting Module** (DTM) helps the astronaut isolate, diagnose, and correct problems in the experimental equipment.
- The **Experiment Suggester** (ES) uses input from the IDF to construct new experiments that investigate previous "interesting" results.
- The **Executive Module** moderates all inter-module communications, and ensures proper and timely allocation of system resources.
- The **Scheduler** monitors the experiment and mission from the macro perspective on the ground and helps the PI plan experiments later in the mission.

These modules are distributed between three computers, two in-flight and one on the ground (not shown in the diagram). The "Data Computer" runs the DAM and DQM, and is connected directly to the on-board experiment computer via an analog-to-digital converter. The back-end "AI Computer" runs the PM, IDF, DTM, ES, and the Executive, and interfaces directly with the astrounaut operator running the experiment. The "Ground Computer" runs the Scheduler and is used by the PI.

Protocol Manager

Protocol: an ordered sequence of steps that guide astronauts in performing an experiment.

DESIGN DESIDERATA:

- The PM should provide advice, not dictate actions.
- The PM should provide tailorable explanations.
- Assumptions should be minimized.
- Astronauts should not have to enter data about progress
- Recommendations should be based on input from the other modules.

Protocol Manager

A protocol is an ordered sequence of steps that guide the astronauts in performing the experiment during a session. A typical protocol may include steps which perform the following tasks: deploy the experimental equipment, set it up, test the apparatus, prepare the subjects, perform the actual experiment on each subject, shut down the experiment, and re-stow the equipment.

The "management" of a particular protocol may consist of adding, eliminating, or altering the order of the steps. The Protocol Manager has the responsibility of monitoring the completion of each step, and will automatically compute modified protocols if the experiment falls behind (or gets ahead of) schedule or encounters interesting data. This module will, therefore, be the most visible to the astronauts and will have the bulk of the user interface requirements.

The following design principles have been followed in the developement of the Protocol Manager:

- The system should provide advice, and let the users take actions.
- The system should provides explanations for the actions it recommends. These explanations should have varying degrees of detail to maximize their usefulness for different users in different situations.
- Interpretations or inferences that require too many assumptions are likely to be wrong, so assumptions should be minimized.
- The system should not force the astronauts to enter information about the progress of the experiment. After all, the system is supposed to simplify the astronauts' task (which is to perform the experiment) not complicate it.
- The Protocol Manager must be able to provide recommendations with no more knowledge about the external environment than what can be inferred from the input provided by the other modules.

Finally, the Protocol Manager must operate in real-time. All calculations that require significant amounts of time must be transparent to the user. In addition, checks must be made to make sure that the assumptions made when a calculation was started are still valid when the results are finally obtained.

Types of Protocols

- Original Protocol
- Modified Original Protocol
- Current Protocol
- Proposed Protocol
- Optimal Protocol
- Protocol History

Types of Protocols

There are several types of protocols:

- the **Original Protocol** that was originally suggested by the PI prior to the mission;
- the **Modified Original Protocol**, a modification to the Original Protocol made during the mission;
- the **Current Protocol**, the protocol that is currently being performed;
- the **Proposed Protocol**, the protocol proposed by the Protocol Manager in response to experimental circumstances (at the option of the astronaut, the proposed protocol can become the current protocol); and
- the **Optimal Protocol**, which includes all steps of interest to the PI, and assumes no time constraints.

In addition, the Protocol Manager maintains a Protocol History, which is a sequence of steps that have been performed already as part of a protocol.

Why Modify the Protocol?

- The experiment is running late, because
  - the session started late,
  - some steps are taking longer than expected,
  - there are problems with the equipment,
  - there are interruptions from Mission Control.
- An astronaut is sick and/or unable to complete the experiment.
- Experimental data is interesting or erratic.

Why Modify the Protocol?

There are several types of events that can occur during the session or the mission that require a change in the protocol that was designed by the PI prior to the mission. The most common event is that the experiment is running late. This may happen for a number of reasons:

- The session started late or some steps are taking longer than expected, but is running fine otherwise.
- There have been problems with the equipment, requiring diagnoses, troubleshooting and/or repair.
- Mission Control interruptions have delayed the experiment. *(Sometimes it is possible to get an extension from the Mission Manager in order to make up for the time lost. The ability to do so will depend on the degree of responsibility of the experiment for the delay and on the nature of the next scheduled activities. If an extension is not granted, or it is not sufficiently long, the protocol needs to be altered.)*
- There is trouble with the equipment or the output signals are abnormal. There is a tradeoff between spending experiment time fixing the problem, or continuing with degraded data.

Other possible events are:

- An astronaut is sick and/or unable to participate in all or part of the experiment.
- The response of a subject during the experiment (or his or her response during previous sessions) is either interesting or erratic. It may be desirable to either add or drop additional runs for that subject.

Protocol Preparation Strategies

- Changes should be modifications of an original protocol.
- Optimize scientific results through "coverage":
  - Get a good "baseline" early in the mission.
  - On mission day zero, get some data on each subject.
  - Early in mission, cover subjects; later in mission, cover test conditions.
  - Stay with the "good data" (e.g. consistent subjects).
  - Improve statistics (i.e. improve the coverage of any one subject, and cover as many subjects as possible).
  - Investigate interesting data immediately.
  - Plan conservatively and prepare alternate plans.
  - Balance science and efficiency; consider both scientific and practical issues.

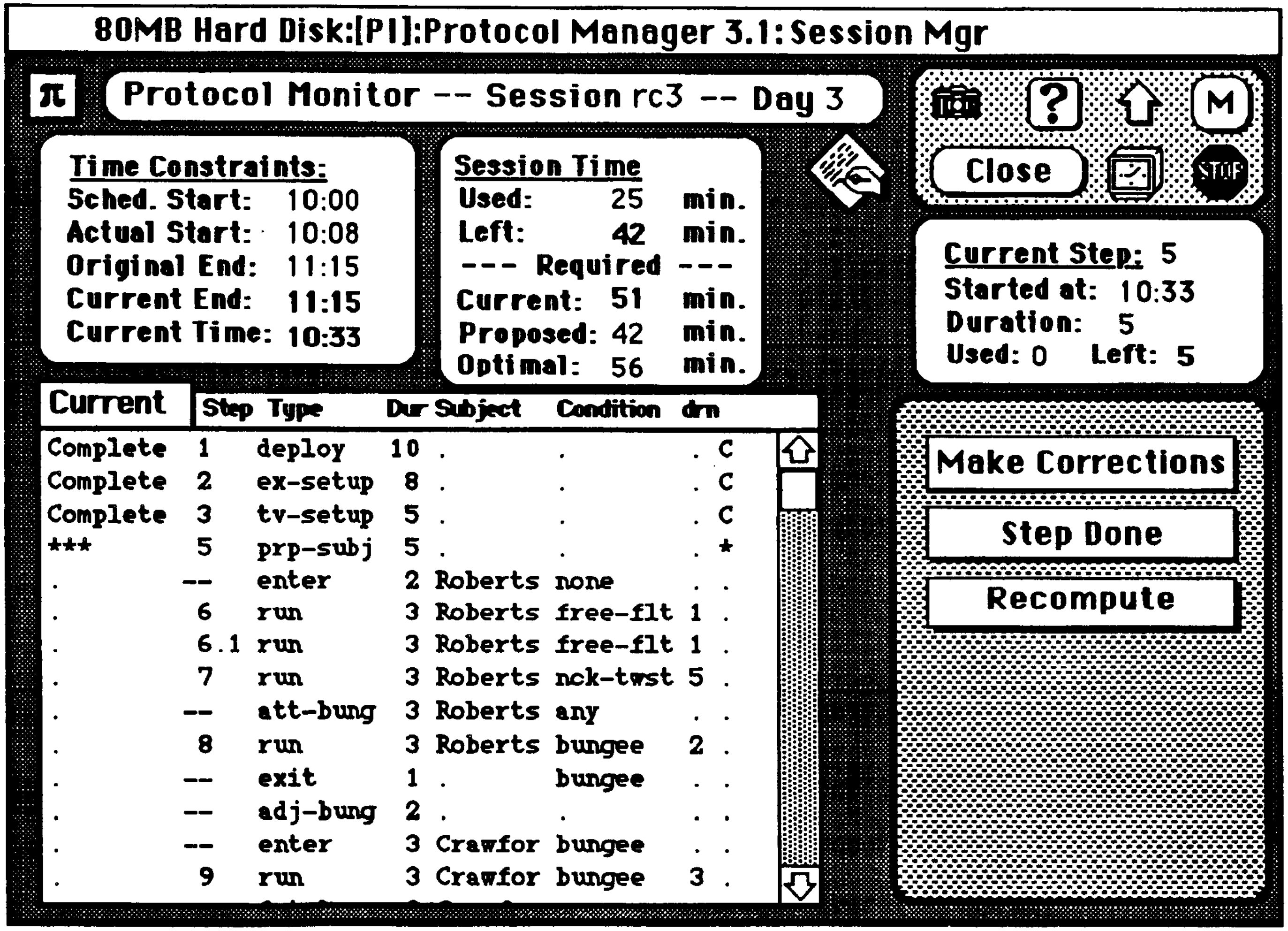
Protocol Preparation Strategies

There are many ways in which a protocol may be assembled., although new protocols are generally not designed from scratch. All changes should be modifications of the protocol originally prepared by the PI. If new experiments need to be included, they are to be suggested by the Experiment Suggester, the PI, or the astronaut.

The fundamental strategy adopted to optimize scientific results is "coverage". This strategy can be described as follows:

- Consider getting a good baseline early in the mission.
- On mission day zero, get at least some data on each subject.
- Early in the mission cover subjects, late in the mission: cover test conditions.
- Stay with the good data (e.g., consistent subjects).
- Improve statistics (i.e. improve the coverage of any one subject, and cover as many subjects as possible).
- Investigate interesting data immediately.
- There are some data signals that are critical at each stage of the mission. Get them.
- Plan conservatively. When time or resources are short, do not assume that requests for them will be granted. Prepare alternative plans, keeping track of when a commitment to a plan is required.
- Balance science with efficiency. From the scientific point of view, do the runs in opposite order for each subject. From a practical point of view, design a protocol that minimizes the time spent setting up subjects or conditions.

PM Interface: Current Protocol



80MB Hard Disk:[PI]:Protocol Manager 3.1:Session Mgr

Protocol Monitor -- Session rc3 -- Day 3

Time Constraints:
Sched. Start: 10:00
Actual Start: 10:08
Original End: 11:15
Current End: 11:15
Current Time: 10:33

Session Time
Used: 25 min.
Left: 42 min.
--- Required ---
Current: 51 min.
Proposed: 42 min.
Optimal: 56 min.

Current Step: 5
Started at: 10:33
Duration: 5
Used: 0 Left: 5

```
Current   Step Type     Dur Subject Condition drn
Complete  1    deploy   10 .        .       . C
Complete  2    ex-setup  8 .        .       . C
Complete  3    tv-setup  5 .        .       . C
***       5    prp-subj  5 .        .       . *
.         --   enter     2 Roberts none     . .
.         6    run       3 Roberts free-flt 1 .
.         6.1  run       3 Roberts free-flt 1 .
.         7    run       3 Roberts nck-twst 5 .
.         --   att-bung  3 Roberts any      . .
.         8    run       3 Roberts bungee   2 .
.         --   exit      1 .       bungee   . .
.         --   adj-bung  2 .       .        . .
.         --   enter     3 Crawfor bungee   . .
.         9    run       3 Crawfor bungee   3 .
```

PM Interface: Current Protocol

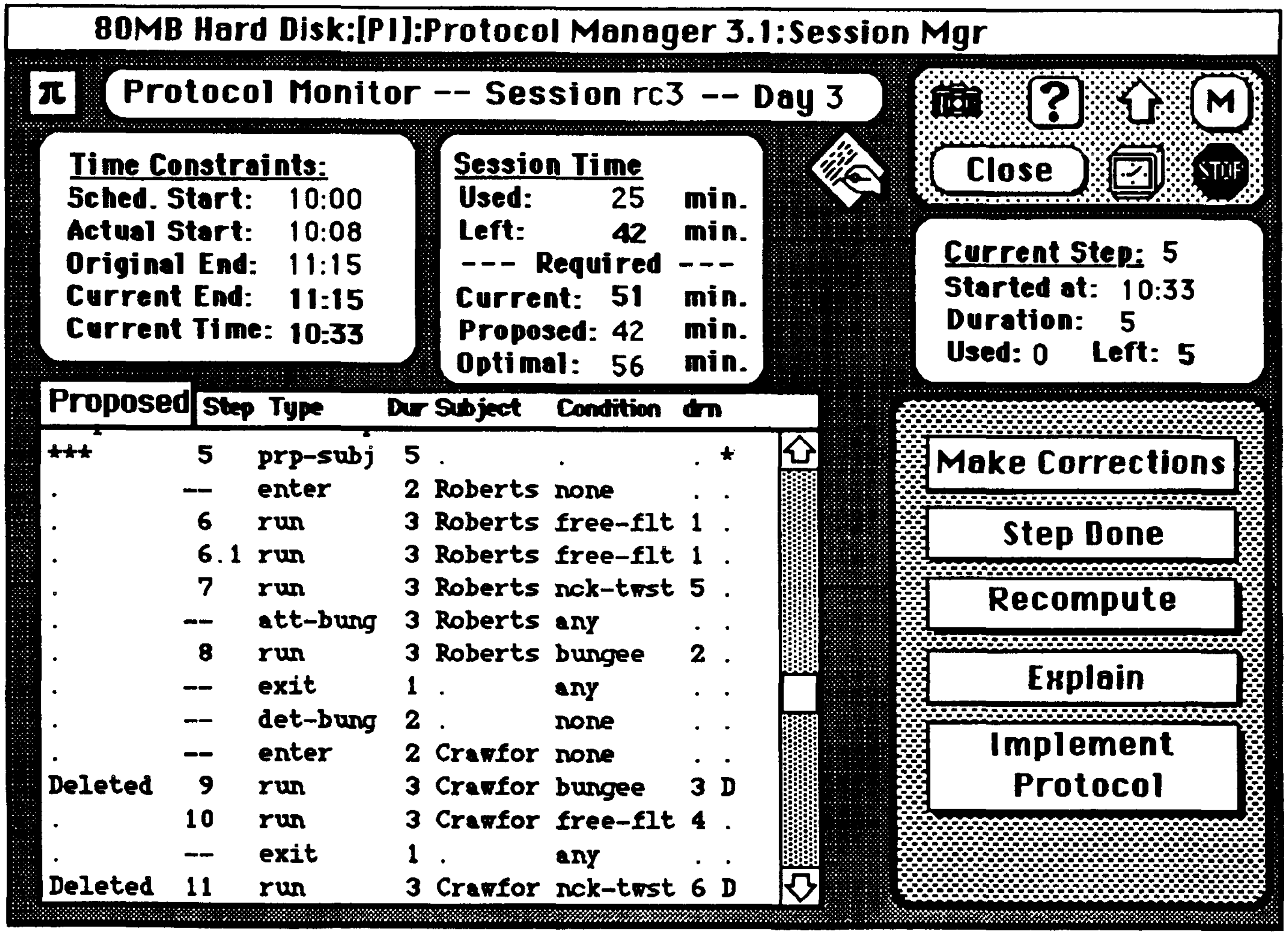
Here we show an example of a Protocol Manager screen. Note that this interface (in Hypercard), at this time serves a dual purpose of prototype user interface and of development interface.

For the developmnet part of the interface many of the buttons are meant to do things such as advancing time or creating interrupt that in the real system would obviously come from other modules.

Work on a user interface for [PI] from the point of view of the astronauts has begun this year at JSC. (Dr. Rudisill's group)

This screen shows that three set-up steps have been completed. More detailed information on each step can be asked of the system if the astronaut needs it.

PM Interface: Proposed Protocol



```
Time Constraints:
Sched. Start:    10:00
Actual Start:    10:08
Original End:    11:15
Current End:     11:15
Current Time:    10:33
```

```
Session Time
Used:        25   min.
Left:        42   min.
--- Required ---
Current:     51   min.
Proposed:    42   min.
Optimal:     56   min.
```

```
Current Step: 5
Started at:  10:33
Duration:    5
Used: 0    Left: 5
```

```
Proposed  Step Type     Dur Subject  Condition drn
***       5    prp-subj  5 .         .         . *
.         --   enter     2 Roberts  none       . .
.         6    run       3 Roberts  free-flt  1 .
.         6.1  run       3 Roberts  free-flt  1 .
.         7    run       3 Roberts  nck-twst  5 .
.         --   att-bung  3 Roberts  any        . .
.         8    run       3 Roberts  bungee    2 .
.         --   exit      1 .        any        . .
.         --   det-bung  2 .        none       . .
.         --   enter     2 Crawfor  none       . .
Deleted   9    run       3 Crawfor  bungee    3 D
.         10   run       3 Crawfor  free-flt  4 .
.         --   exit      1 .        any        . .
Deleted   11   run       3 Crawfor  nck-twst  6 D
```

PM Interface: Proposed Protocol

The experiment is late: 42 minutes are left and the current protocol would require 51 minutes (see "Session Time" box).

Note that this protocol suggests the deletion of steps #9 and #11. The astronaut could ask for the reasons for these choices.

Typical reasons might be that a particular subject has been giving erratic data or that some other step appears to be more important. Note for instance that step 6.1 is a repetition of step 6, yet it has been inserted in spite of lack of time probably because that step, in a previous session, had produced results that were deemed "interesting" i.e. in need of confirmation.

Interesting Data Filter

- **Discovery**:

  The observation of some unexpected phenomenon with the consequent revision of a previous theory or model.

- **Goal of the IDF**:

  To provide an extensible mechanism for automated discovery that is useful to the astronaut and thus the PI.

Interesting Data Filter

One of the major constraints of present experimentation in space is a fixed protocol. Depending on new contingencies that may arise the system will be able to present the astronauts with a new course of action, and with an explanation for any new suggestions.

A very important type of possible contingency is finding that some parameters that are being observed seem to have more or less significance than had been anticipated. In either case, being able to plan accordingly and without delay greatly improves the quality of experimentation, by enabling the astronauts to focus more on work that is likely to be significant.

One of the system modules is being constructed to provide this kind of information about the data that are being observed. Data will be flagged as interesting on the basis of discrepancies from previous experience or relevance to alternative models of phenomena under study.

Some of the greatest discoveries in science have happened as a result of observations that had not been planned at all, when an unexpected phenomenon caught the eye of the researcher. The ability to contribute to this type of discovery is at the forefront of AI research and we are planning to use this system as a testbed for any promising approaches in this this area.

The "Interesting Data Filter" (IDF), as a long term goal for PI-in-a-box is the most advanced aspect of the whole project. The goal of the project is to do better science in an environment where the PI cannot be present and usually cannot have up-to-the-minute information. Certainly the essence of science is "discovery", i.e. the observation of some unexpected phenomenon with the consequent revision of a previous theory or model. The ability to make discoveries and revise one's own world model can be viewed as the essence of intelligence as well, thus clearly a long term goal of artificial intelligence. Unfortunately, present AI technology is still a long way from providing a framework for serious discovery capability. Our position is to keep this long term goal in mind and to begin providing step by step whatever discovery-related capability could be useful to the astronaut and/or the PI. The first necessary capability is to determine whether data is "interesting". We define below what we mean by this.

What is "Interesting Data"?

- Unanticipated events or occurrences
- Parameter values produced during the experiment that are different from those expected
- In general, data that is "in need of confirmation"

What is "Interesting Data"?

The preliminary development of the IDF quickly revealed that "interesting" meant different things to different people, or even to the same person at different times. Parameter values that were expected, but previously never confirmed could be viewed as interesting, as well as values that were unexpected, and, in both cases, once established, they would cease to be interesting. We settled on the concept that interesting corresponded to "in need of confirmation" and that the most common reason for such need was the finding that the values produced by the experiment were different from those expected. From this point of view, data that have been confirmed cease to be interesting.

As a consequence of confirming unexpected data, researchers are in general led to revise the reasons behind their original expectation, and to build a new set of expectations that will need confirmation. Typically the reasons behind a set of expectations are embodied in a quantitative or qualitative model of the phenomenon under observation. We envision that, at least for some parameters, the IDF will be able to aid the researcher in such a revision cycle, by explaining the reasons for the original expectations and by proposing consequences for the model of the discrepancies encountered.

Strategies for Discovery

- Facilitate the construction and revision of predictive models.
- Facilitate the observation of parameter relationships.
- Examples of discovery strategies that can be employed:
    - If a parameter is interesting, suggest observing a "related" parameter.
    - Diagnose problems with the present model.
    - Suggest new hypotheses and relationships.

Strategies for Discovery

At this time the IDF only compares parameter values against expected values for the same parameter. The next step will be to handle cases where the relationship between two different parameters might not be what was expected. For instance our model might have predicted that a parameter in condition A should be greater than in condition B but the opposite is found instead, or no effect. Even though the present model might not be sufficiently predictive for the values of a specific parameter, it may be predictive of some relationships. This will be the next step in the IDF evolution.

So far we have only considered parameters that the experiments were designed to measure. An exciting aspect of scientific discovery is the chance of noticing phenomena that were completely unexpected. We have a long term goal to address this problem. Our first approach will be to examine strategies normally employed by researchers. For instance, if a parameter gives unexpected values can we think of a related parameter that could show the same effect? In this case the system, after having suggested and obtained the repetition of a run for the purpose of confirming the unexpected value, would suggest testing the "related" parameter to begin exploring the implications of the finding. The more predictive a model we can build, the richer the discovery strategies we can employ.

Data Analysis and Quality Monitoring

- **DAM** - Data Acquisition Module
  - Acquires raw data from experiment equipment
  - Extracts meaningful parameters: means, standard deviations, peak values, trends
  - Looks for data indicating specific events
- **DQM** - Data Quality Monitor
  - Checks for dead, erratic, or suspiciously steady signals
  - Notifies Executive Module when data is questionable
- **DTM** - Diagnosis and Troubleshooting Module
  - Invoked when data seems abnormal
  - Interacts with astronaut to diagnose and perhaps repair equipment problems
  - Inhibits analysis and interpretation when data is bad

Data Analysis and Quality Monitoring

The DAM and DQM modules perform real-time acquisition, testing and analysis of experimental data. Such activity is both CPU intensive and memory intensive. It was therefore decided to integrate them in the same piece of software running on a dedicated machine. In order to avoid any disturbance of the existing experimental hardware, the signals of interest are branched off the existing experiment computer and digitized into a data acquisition and processing software package called LabVIEW.

We are currently developing a software layer which extracts various parameters of interest to the PI and therefore to its substitute [PI]. These include means, standard deviations, peak values, trends, occurrences of specific events, etc. These values are calculated for each trial and sent over once every run.

The same software also performs data quality checks. This module looks for dead signals, erratic signals or suspiciously steady signals. A warning message indicating the signal and the kind of problem is sent to the second machine for intelligent analysis after each trial. In some cases, the scheduling module calls for a delay in the transmission of the error message until the next trial or the termination of the execution of the current module.

In case of an abnormal signal, DTM is called and decides, with possible override by the astronaut, to perform troubleshooting and/or repair, or simply to disregard a faulty signal as of secondary value. The DTM will be implemented as an expert system, often requiring interaction with the astronaut. It is responsible for telling other modules, such as IDF or PM about the validity of the information extracted from the various signals. The DQM is purely algorithmic and makes no decisions. If DTM decides to discard a signal, DQM will still recognize it as faulty and the results will be discarded downstream. This allows for a clear separation of the real-time number intensive tasks and the "intelligent" processing of data. It also alleviates the need for two-way communications between the machines and allows for unexplained auto-repairs.

The system will monitor incoming data from the experiment in order to detect potential problems with the instrumentation. If a problem is detected, the astronaut is notified and, if appropriate, aided in the trouble-shooting process. Whether trouble-shooting is an appropriate course of action may depend on other constraints such as available time, importance of the problem signal and other considerations.

User Interface Issues

- **The astronauts are in control.**
  - Allow astronauts to override recommendations and/or correct current assumptions.
  - Allow astronauts to exert direct control on the Protocol Manager.
  - Permit astronauts to explore "what if" situations.
- **Minimize reliance on astronaut input.**
  - Infer rather than ask (when possible).
  - Facilitate rollback, recovery, and re-processing.
  - Always display status and progress information.
  - Avoid displaying annoying information.

User interface issues

We have made specific assumptions about how the system should interact, functionally, with the astronauts. The most important assumptions are the following: a) The astronauts know the most about the events of the moment surrounding the experiment. b) They should always be able to override any recommendations and/or correct the current assumptions. c) The astronauts must be able to exert a fairly direct control on the operation of the Protocol Manager through the ability to force or inhibit certain steps (including runs) of the protocol. This includes the ability to request double runs and change the order of steps or blocks of steps (these principles have not been fully implemented). d) Minimize reliance on astronaut-provided data to infer the current state of the protocol. Infer as much as possible of the current state from the signals of the other modules, without making overly complex assumptions. e) Rollback and re-processing may be necessary. It is easy for the astronauts to make mistakes during a session and execute the wrong steps (by pushing the wrong button for instance). It is very important for them to know how to recover. These mistakes may not be noticed until sometime later. f) The astronauts should be able to see the alternative protocol proposed by the Protocol Manager, on demand. The changes with respect to the current protocol should be highlighted. g) Avoid annoying information such as constant protocol changes that have no effect on the current step, or notices when there are no changes required. h) Let the astronauts be aware of what is going on. For instance, let them know that a background computation is in progress and the reason for it. i) Allow the astronauts to explore "what-if" situations. Of particular interest is the ability to move the clock forward (this has not been implemented).

Both for the Protocol Manager and the IDF, the actual implementation of the screens has been done in Hypercard for a mixed purpose: to provide a development interface and to illustrate the underlying user interface assumptions and system capabilities to evaluators and to prospective users. We are now beginning to give serious consideration to the human factors involved in the actual presentation of data to the astronauts (work begun in Dr. Rudisill's group at JSC).

Explanations

- **Protocol Manager**
    - Increase familiarity with system operation
    - Enhance confidence in the system and understanding of the experiment
    - Three levels of explanation:
        - what is being accomplished
        - what the trade-offs are
        - detailed justification for protocol modification
- **Interesting Data Filter**
    - How conclusions are derived from numeric data
    - How qualitative and quantitative expectations were formed
        - model-based considerations
        - details of expectation formulation
    - How experimental data differs from expectations

Explanations

Explanations are a major part of the design of the user interface. The astronauts may have their own suggestions on how to improve the efficiency of the experiment. These may be different from what the Protocol Manager proposes. Proper explanations may allow them to evaluate the relative merits of the alternatives. Explanations provide a way for astronauts to become familiar with the operation of the system and the issues associated with protocol management during ground training sessions. This will enhance their confidence in the system and their own understanding of the experiment. Explanations should be designed to be understood quickly and selectively. Three levels of explanation are required: a) What is being accomplished. This may be implemented by showing the significant circumstances that led to the proposed protocol, and how they are different from the previous recommendation. Work still needs to be done in this area. b) What is the trade-off. This may be implemented by showing the insertions and deletions on the proposed protocol with respect to the current one. c) What are the detailed reasons. A detailed account that led to the inclusion (or exclusion) of a particular step should be available.

Explanations must also be produced by the IDF and the reasons are similar to those assumed for the Protocol Manager. Four different types of explanation are being considered: a) High level strategic. This states how a value differs from what was expected. b) Lower level strategic, an explanation of how the expected value was determined. c) High level model related. What considerations about the physiological model led to a particular qualitative expectation. d) Lower level model related. Detailed, model related steps, for how a particular qualitative expectation was reached. Strategic explanations deal with how conclusions are gleaned from numeric data and do not require an understanding of the underlying physiology. Model related explanations rely on an understanding of the underlying physiology and will constitute the basis for further exploration of new model assumption and new parameters. Some problems in this area were discussed above in the section on the IDF.

Conclusions

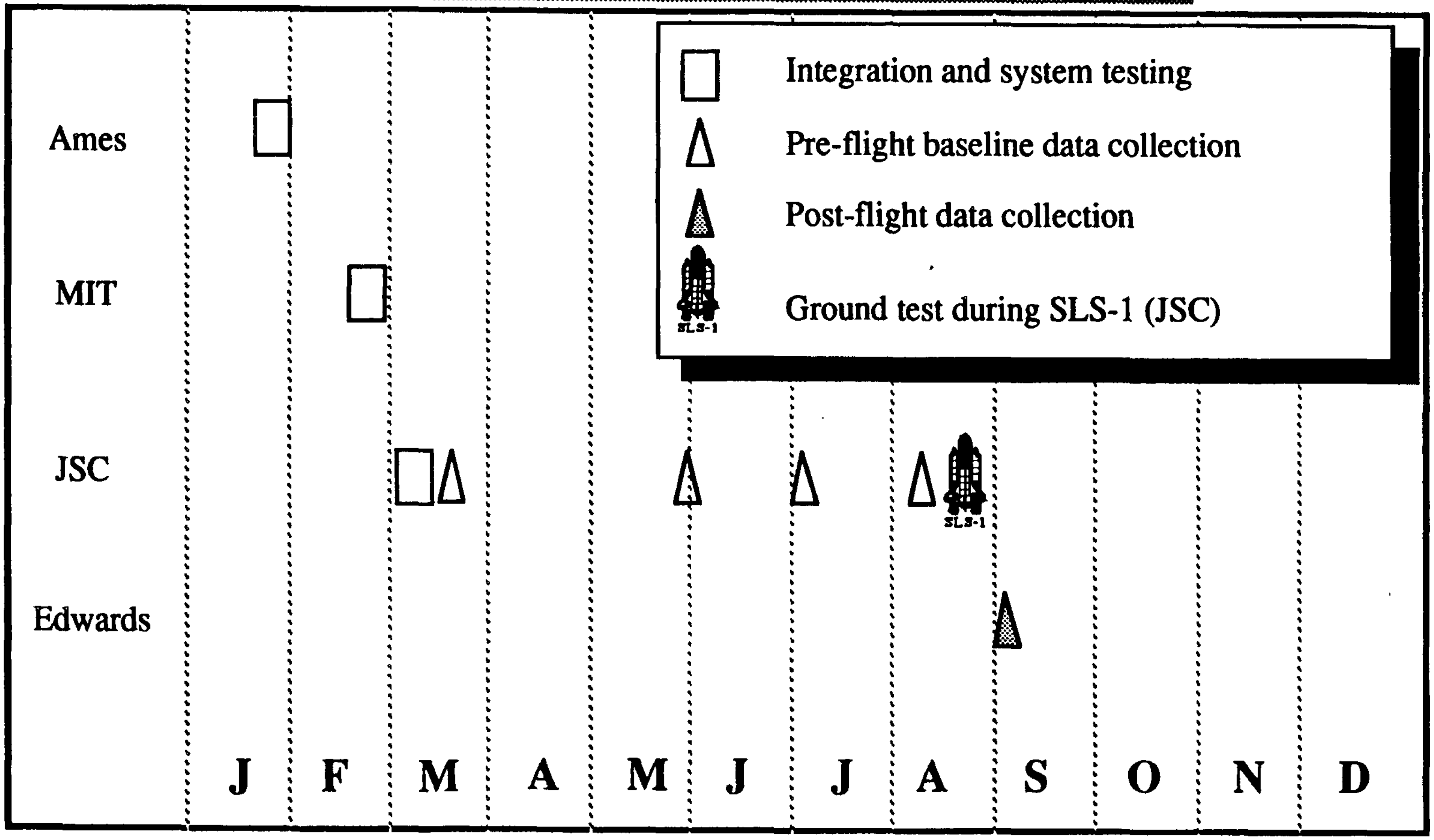
- AI techniques can significantly improve astronauts' ability to perform in-flight science.
- PI-in-a-box, although experiment-specific, addresses issues of importance to many classes of experiments:
  - protocol flexibility
  - detection of interesting phenomena
  - user interface issues
  - real-time data acquisition and monitoring
  - troubleshooting of experiment equipment

Conclusions

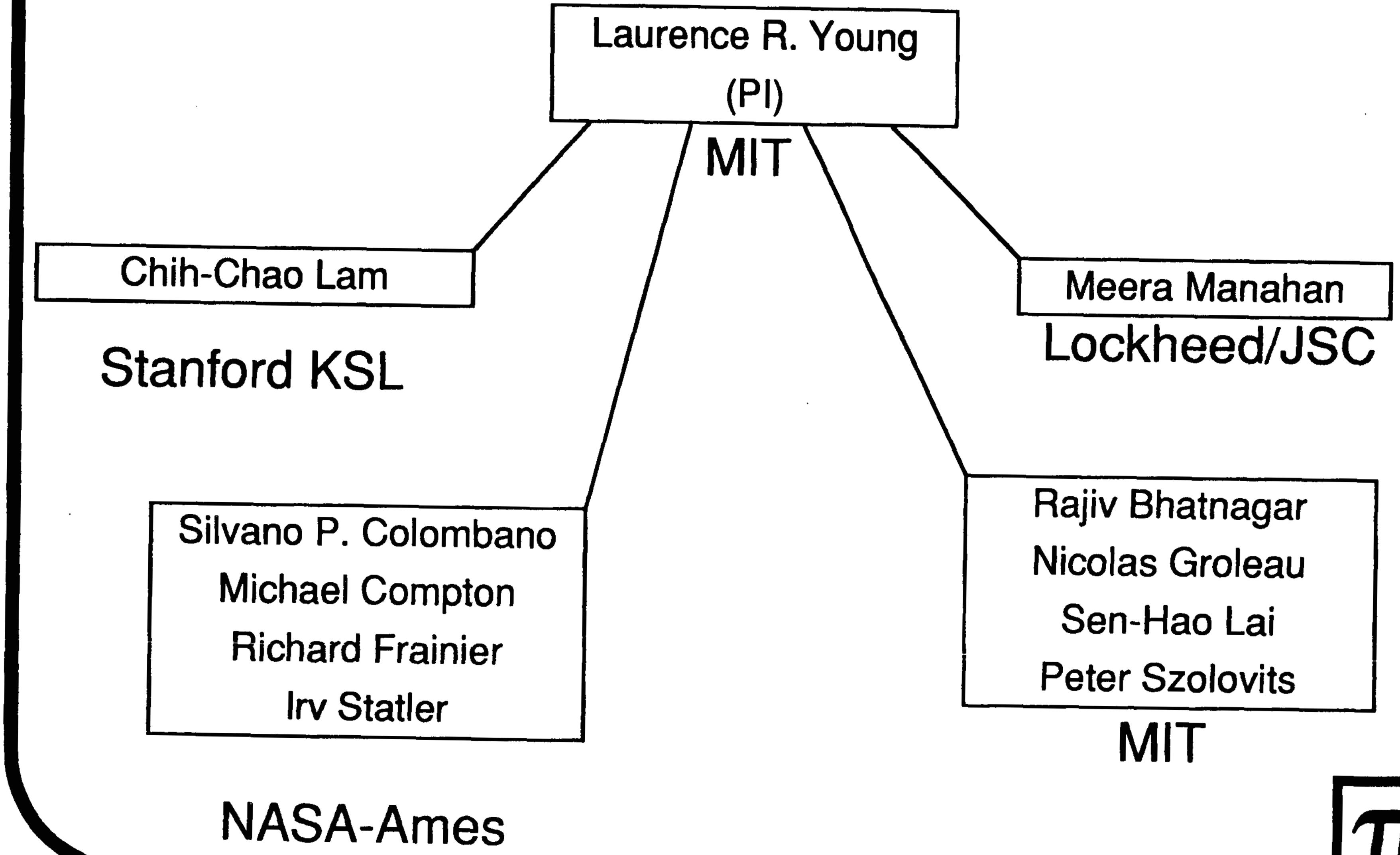
We have presented a broad description of our work on an expert system designed to improve the quality of space experimentation in vestibular physiology. It appears that the issues raised by this work, such as protocol flexibility, detection of interesting phenomena, user interfaces and real-time data acquisition and monitoring have important implications for space experimentation in general.

So far we have concentrated on the Protocol Manager, the Interesting Data Filter, the Data Acquisition and Data Quality modules, which have been developed separately. Only the last two modules and have been hooked up to a real data stream. Integration of the whole system is a major goal of FY '90.

Milestones for 1990



Project Team



Laurence R. Young
(PI)
MIT

Chih-Chao Lam
Stanford KSL

Meera Manahan
Lockheed/JSC

Silvano P. Colombano
Michael Compton
Richard Frainier
Irv Statler
NASA-Ames

Rajiv Bhatnagar
Nicolas Groleau
Sen-Hao Lai
Peter Szolovits
MIT

Milestones

January '90: First integration test

March '90: First BDCF test

April '90: 2nd BDCF test

August '90: Test of ground support for
experiment on SLS-1

August '92: Test of flight support for
experiment on SLS-2

Johnson Space Center-Houston, Texas

FLIGHT SYSTEMS
AUTOMATION

SYSTEMS DEVELOPMENT & SIMULATION DIVISION

LESC/J. TRAN
EF5/H. HIERS

FEB. 7, 1990

REMOTE MANUPILATOR SYSTEM (RMS) AUTOMATION

RMS Expert System - Background

This is a joint project between Johnson Space Center (JSC) and Ames Research Center (ARC) to extend OAST-sponsored Procedural Reasoning effort to the orbiter subsystem diagnosis and control problems. A Reaction Control System (RCS) procedural expert system was developed and demonstrated to ARC and JSC representatives in 1987. The emphasis of the RMS Expert System project is to demonstrate a dynamic, real-time procedural reasoning expert system integrated with the Systems Engineering Simulator (SES).

BACKGROUND

- JOINT PROJECT BETWEEN JOHNSON SPACE CENTER AND AMES RESEARCH CENTER TO EXTEND PROCEDURAL REASONING RESEARCH TO ORBITER SUBSYSTEMS DIAGNOSE/CONTROL PROBLEMS
- FUNDED BY THE NASA STRATEGIC PLANS & PROGRAMS DIVISION OF SPACE STATION (CODE ST)
- PROCEDURAL REASONING SYSTEM WAS DEVELOPED AT SRI INTERNATIONAL

RMS Expert System - Objectives

The objectives of the project are to be accomplished over a three-year period. The first year will be devoted to the acquisition of the PRS tool, and the development of a prototype RMS expert system including the knowledge base, a Crew Interface, a RMS software simulator, and defined malfunction scenarios. In the second year, the interface to the SES and the Data Acquisition and Monitoring Module will be built, and the RMS Expert System will be integrated with the SES for evaluation and demonstration to RMS experts. Recommended changes will be incorporated, and the system re-evaluated in preparation for the development of the specifications for an ADA-based procedural reasoning shell in the space station SSE environment.

OBJECTIVES

- EVALUATE THE PROCEDURAL REASONING SYSTEM BY DEVELOPING A RMS DIAGNOSTIC EXPERT SYSTEM TO BE INTEGRATED WITH THE REAL-TIME SYSTEMS ENGINEERING SIMULATOR (SES) AT JSC

- DEVELOP SPECIFICATIONS FOR AN ADA-BASED PROCEDURAL REASONING TOOL FOR EMBEDDED REAL-TIME APPLICATIONS IN THE SOFTWARE SUPPORT ENVIRONMENT OF SPACE STATION

RMS Expert System - Approach

The Shuttle Remote Manipulator System was selected as the domain in which to assess the capability of the PRS shell for real-time fault diagnosis. Because of the time constraint, many tasks were performed concurrently, e.g. knowledge acquisition and encoding, development of a user-friendly color interface, and development of a miniature RMS simulator for testing and debugging. Selected malfunction scenarios defined the scope of the initial integrated prototype.

With technical support from SRI, and through the help of a domain expert, RMS operational and malfunction procedures were implemented as PRS KAs. These procedures were taken directly from the RMS malfunction procedures document, and represent the core declarative knowledge of the expert system. A RMS database was implemented to represent the current beliefs or facts about the domain, and a RMS software simulator was developed for systems testing and debugging.

APPROACH

- WORK WITH RMS DOMAIN EXPERTS FOR KNOWLEDGE ACQUISITION AND PROBLEMS DEFINITION

- PRS TRAINING AND ENCODING OF RMS PROCEDURES FROM FLIGHT DATA FILE

- DEVELOP RMS SIMULATOR AND CREW INTERFACE PROTOTYPES

RMS Expert System - First year achievements

The RMS expert system was prototyped and demonstrated to RMS experts and sponsors. The Crew Interface and the PRS shell hosting the RMS KB and database were integrated. About 22 percent of the applicable RMS malfunction procedures and their associated nominal procedures were implemented. RMS domain experts contributed significantly in defining malfunction scenarios, identifying areas of interest to the RMS astronauts, and evaluating the crew interface. The RMS simulator was completed and tested, but has not been integrated with the expert system. Intended for initial testing and evaluation, the RMS simulator will eventually be replaced by the real-time high-fidelity JSC SES.

Johnson Space Center-Houston, Texas

FLIGHT SYSTEMS AUTOMATION

SYSTEMS DEVELOPMENT & SIMULATION DIVISION

LESC/J. TRAN
EF5/H. HIERS

FEB. 7, 1990

FIRST YEAR ACHIEVEMENTS

- PRS TRAINING BY SRI INTERNATIONAL
- KNOWLEDGE ACQUISTION AND ENCODING
- RMS SIMULATOR DEVELOPED
- RMS CREW INTERFACE DEVELOPED
- SYSTEMS INTEGRATION AND DEMONSTRATION

BACKGROUND

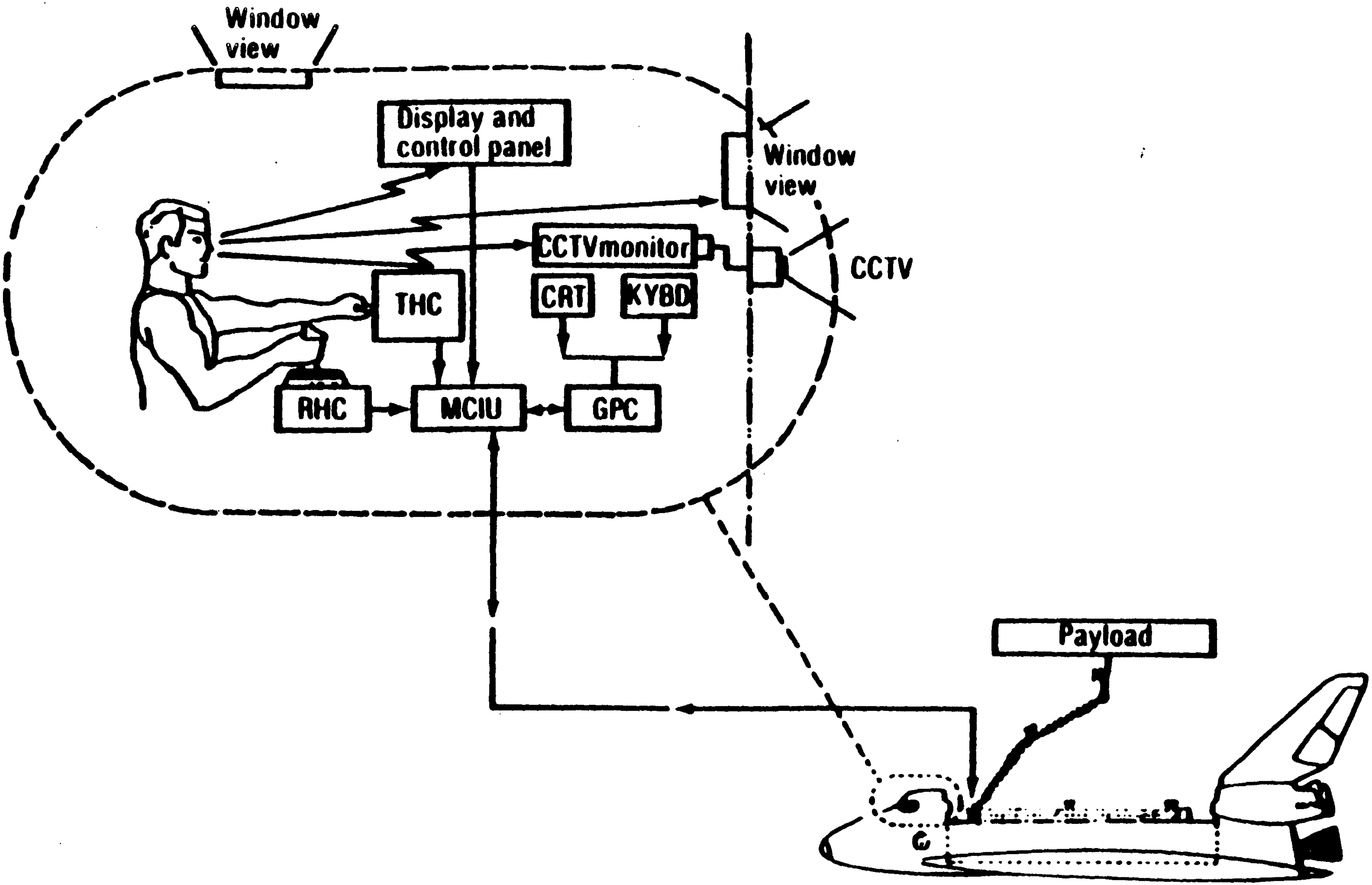
- The RMS was selected as the initial candidate for the application of the Procedural Reasoning System (PRS)

- Specifically, PRS is evaluated with the End Effector (EE) Malfunction Procedures

- The main focus is on the interpretation of the EE status indicators which are located on the RMS Displays and Controls panel

Shuttle RMS

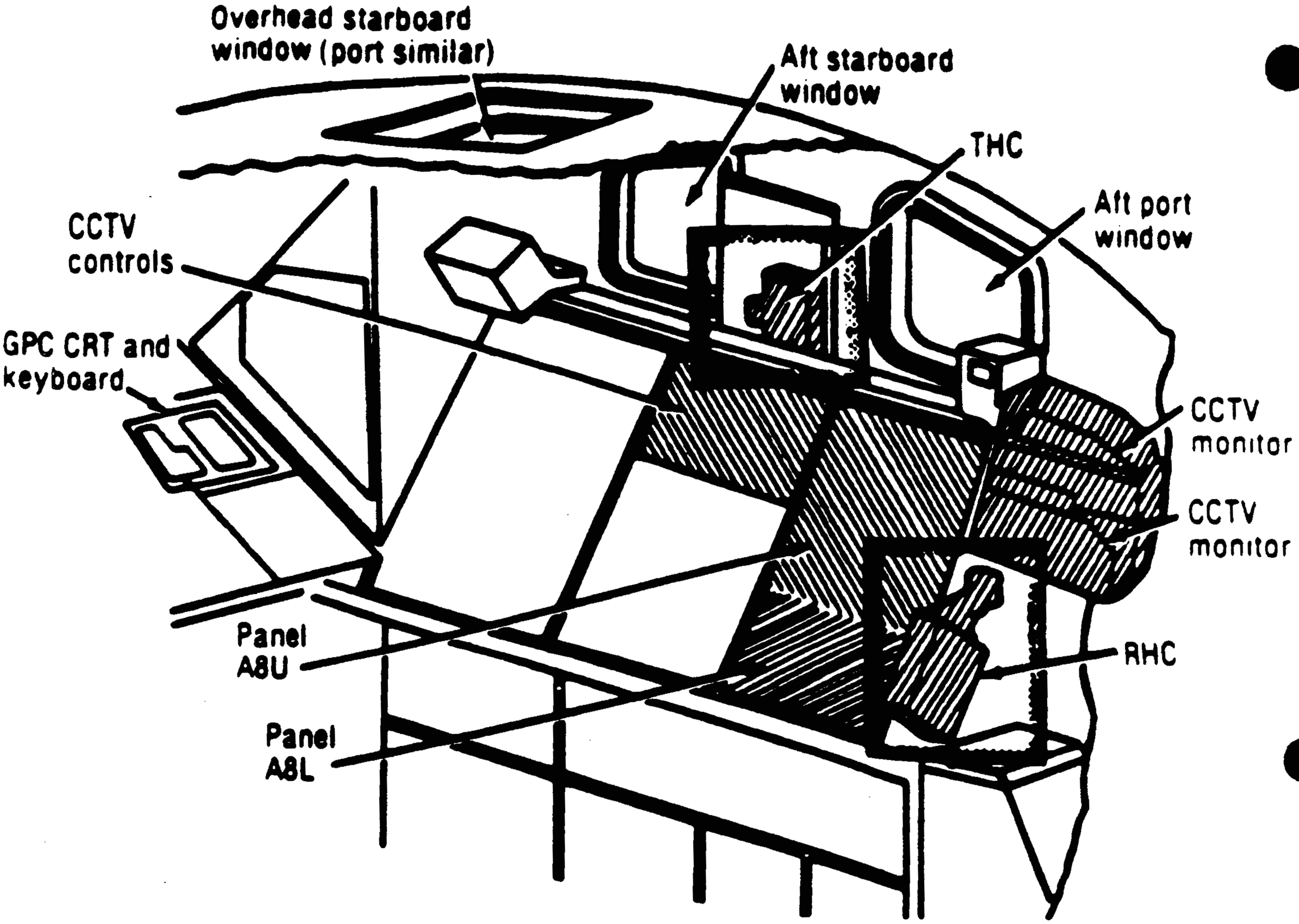
This slide shows the configuration of the aft cockpit RMS controls. The hand controllers, THC (Translational Hand Controller) and RHC (Rotational Hand Controller), and the display and control panel are used to control and monitor the RMS, while the MCIU (Manipulator Controller Interface Unit) provides communication of the GPC (General Purpose Computer) computer with the RMS, informing the computer of all sensor data and relaying all control information to the arm. Visual contact of the payload and the RMS is imperative, so windows and two black and white closed circuit TV (CCTV) monitors are available in the area where the RMS controls are located to allow the user to have greater control of the RMS operations. Each monitor is capable of having a split screen and has zoom capabilities.


Window view
Display and control panel
Window view
CCTV monitor
CCTV
THC
CRT
KYBD
RHC
MCIU
GPC
Payload

Shuttle RMS

RMS displays and controls, aft station

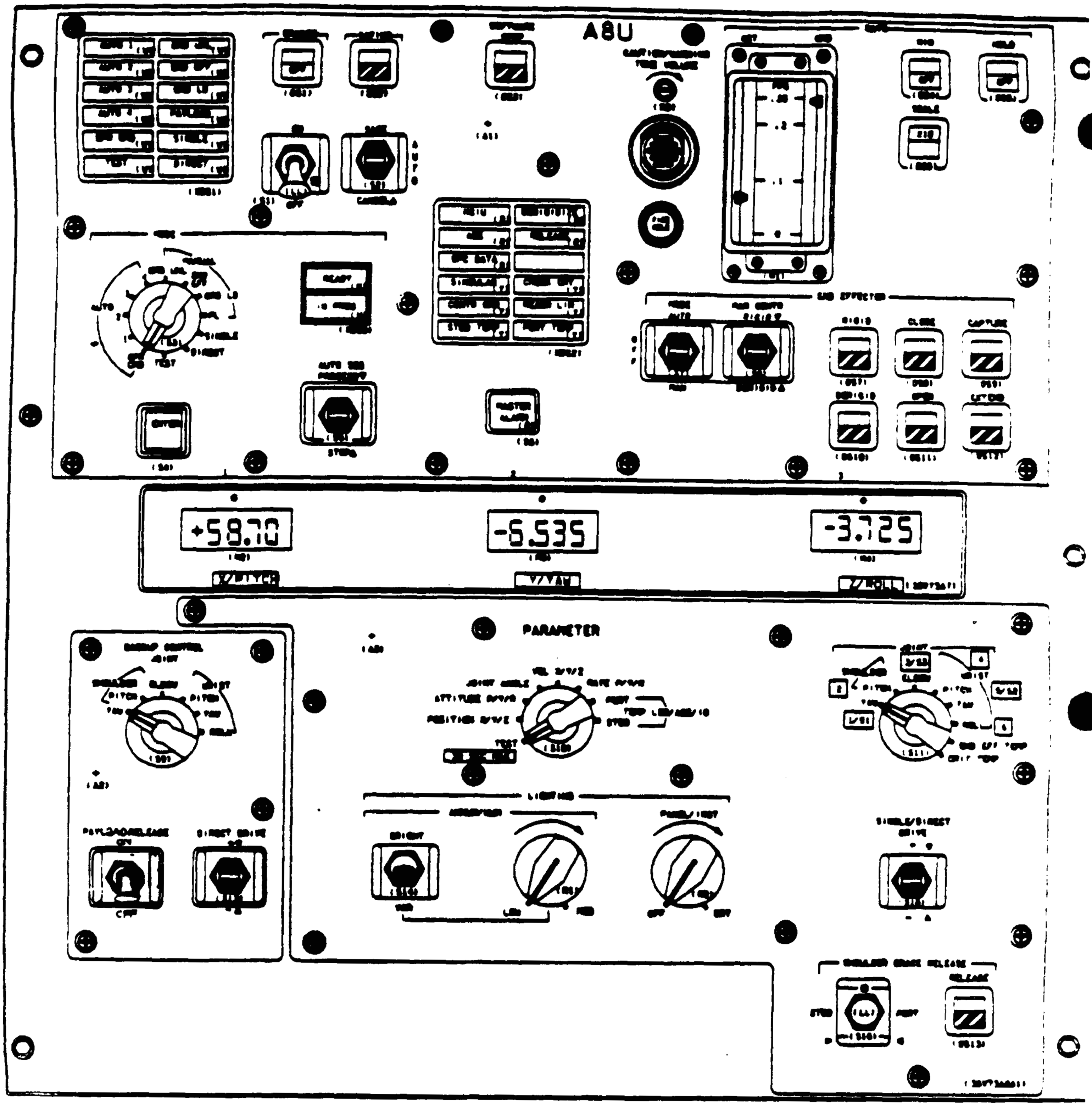
The RMS astronaut operates the RMS from the orbiter aft-cockpit where the RMS controls, viewports, and monitors for are located. The two hand controllers and the main display and control panels, A8U and A8L, are located in the vicinity of the aft-cockpit port window and the CCTV monitor to facilitate the use of the port RMS. In addition the CCTV can be controlled to show one of the 7 possible views using the CCTV controls.



RMS displays and controls, aft station.

Panel A8U

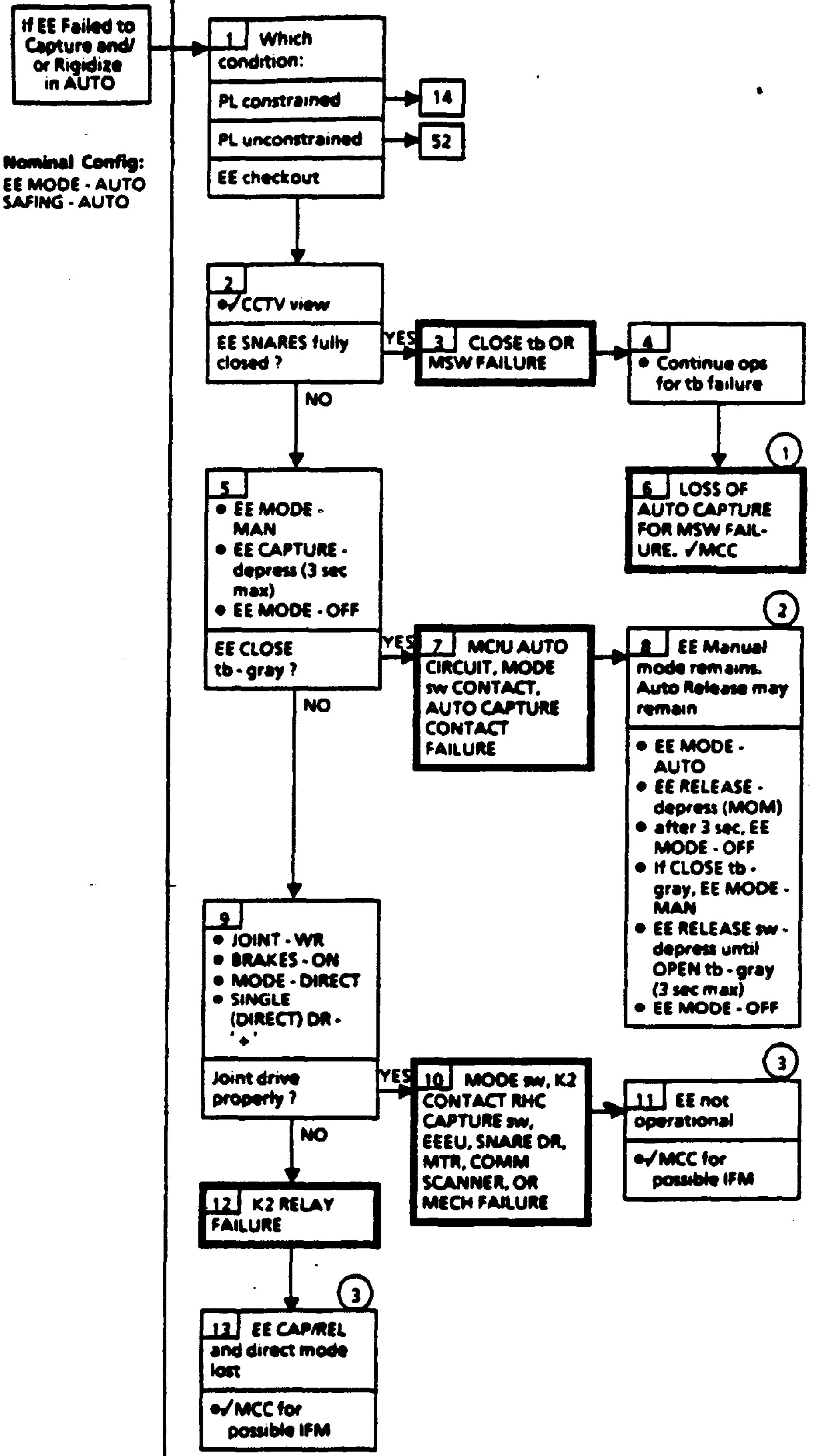
The A8U panel is one of the primary display and control panels of the RMS in the aft cockpit. In the upper left is the mode controller which controls the operating mode of the RMS. To the right is the RATE indicators which indicates the rate of the of the speed of the point of resolution translation. Below this is the end effector switches and talkbacks which indicate the status of the end effector and allow the user to switch between manual and auto control. In the upper center of the screen, the RMS light matrix indicates any problems with the RMS as they occur. The lower portion of the panel is used to transfer control to the various joints of the RMS.


A8U
+58.70
-6.535
-3.725
X/PITCH
Y/YAW
Z/ROLL
PARAMETER
JOINT ANGLE
ATTITUDE
POSITION
TEST
LIGHTING
SINGLE/DIRECT
DRIVE
SHOULDER BRACE RELEASE
RELEASE
AUTO
SINGLE
DIRECT
TEST
READY
MASTER ALARM
OFF

Panel A8U.

13.3a EE FAILS TO CAPTURE/RIGIDIZE IN AUTO

Nominal Config:
EE MODE - AUTO
SAFING - AUTO



① MCC can see microsw status

② Limping will not occur in EE Manual for Auto Capture contact failure

③ RMS D&C IFM KIT available to recover loss of K1, K2 relay, EE MODE sw, RHC capture/release sw, EE MAN CONTR sw. Refer to RMS CONTINGENCY OPERATION/INSTALLATION AND REMOVAL OF THE RMS D&C IFM KIT (IFM) and RMS SSR-4, RMS IFM D&C KIT

EE MALFUNCTION PROCEDURES EXECUTION

- **Failure detection:** noting abnormal EE status indicator(s), caution & warning light
- **Quick response:** immediate safing of the RMS
- **Procedure selection:** searching for the appropriate procedure in the Malfunction book
- **Procedure execution:** performing block by block until a terminal block in the flow chart is reached
- **Further assistance:** contacting MCC as necessary

NEEDS FOR AUTOMATION

- Failure detection - saves the crew from constantly monitoring multiple systems
- Quick response selection - supports the crew in evaluating the immediate condition
- Quick response recommendation - provides the crew with a response to the immediate condition
- Quick response execution - offers the crew an option to perform the response automatically
- Response playback - allows the crew to review data for analytical or training purposes
- Response reasoning - benefits the crew with detailed explanations of the quick response taken

NEEDS FOR AUTOMATION (CONTINUED)

- Procedure selection - supports the crew in evaluating the failure after the system is safed
- Procedure recommendation - provides the crew with the steps to diagnose the problem
- Procedure execution - offers the crew an option for an automatic diagnosis
- Procedure playback - allows the crew to review the diagnosis for analytical or training purposes
- Procedure reasoning - benefits the crew with detailed explanations of the steps taken in the diagnostic procedure
- Failure documentation - assists the crew in tracking failures accurately

NEEDS FOR AUTOMATION (CONTINUED)

- Failure simulation - accommodates the crew during training in real-time
- Procedure upgrade - permits the crew to modify the procedures, the logic, and the commands in real-time
- Procedure verification - presents the crew with a systematic method to validate the procedures in real-time

RMS Expert System - software

The RMS Expert System is being developed incrementally, and involves the implementation of many software modules residing on the color Symbolics Lisp machine. The Lisp environment coupled with the PRS shell provides for a rich development environment that is ideal for the selected problem domain. The goal is to integrate the RMS expert system with the on-orbit simulation test-bed, the real-time high-fidelity JSC Systems Engineering Simulator hosted by the Gould computers. The Data Acquisition & Filtering Module (DAFM) will read real-time RMS telemetry data from the SES, and deliver them to the PRS shell which hosts the RMS KAs (procedures) and database. Concurrently, the crew interface module will interact with the RMS crew interface via communication messages.

Johnson Space Center–Houston, Texas

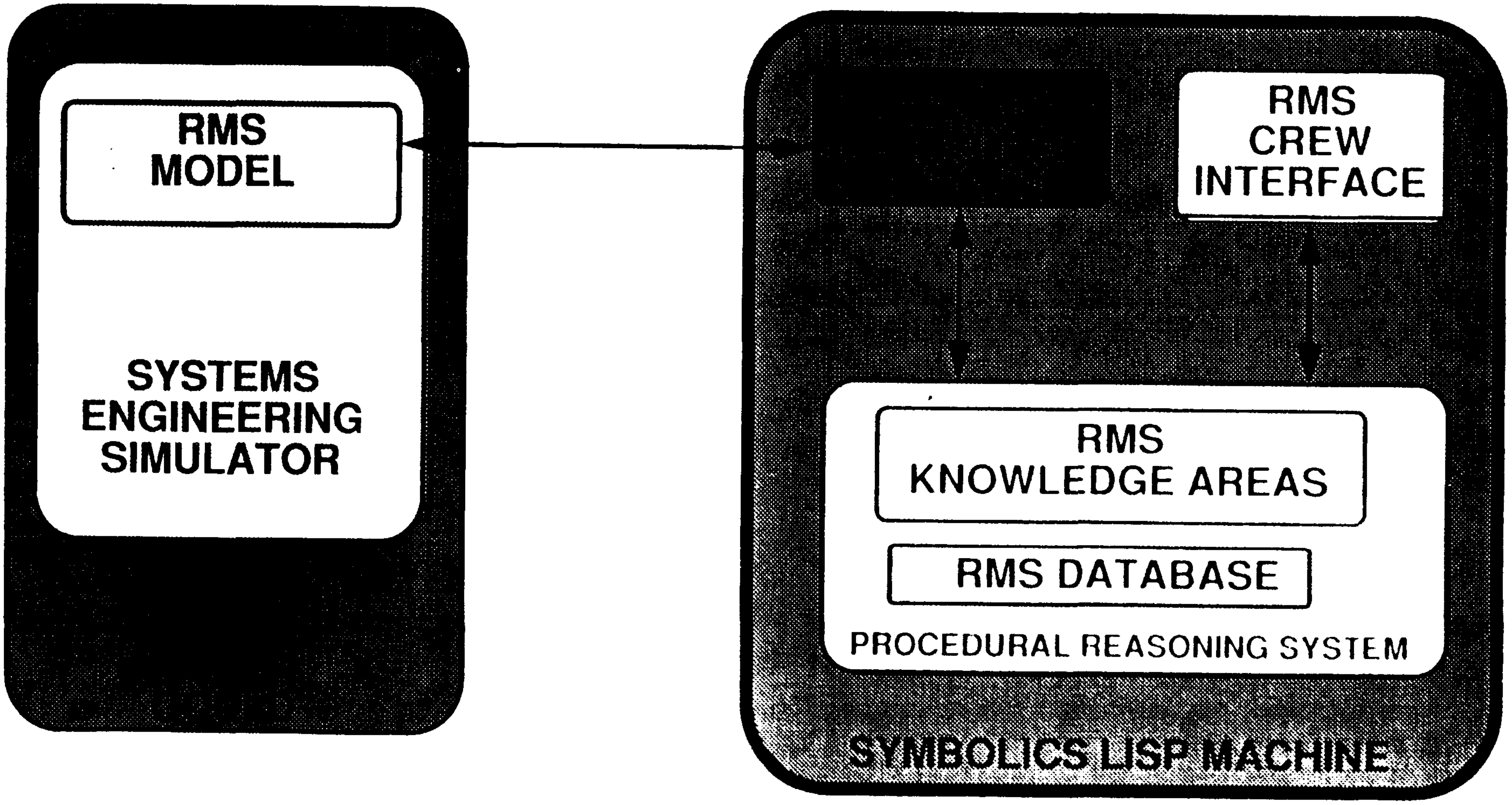
FLIGHT SYSTEMS AUTOMATION

SYSTEMS DEVELOPMENT & SIMULATION DIVISION

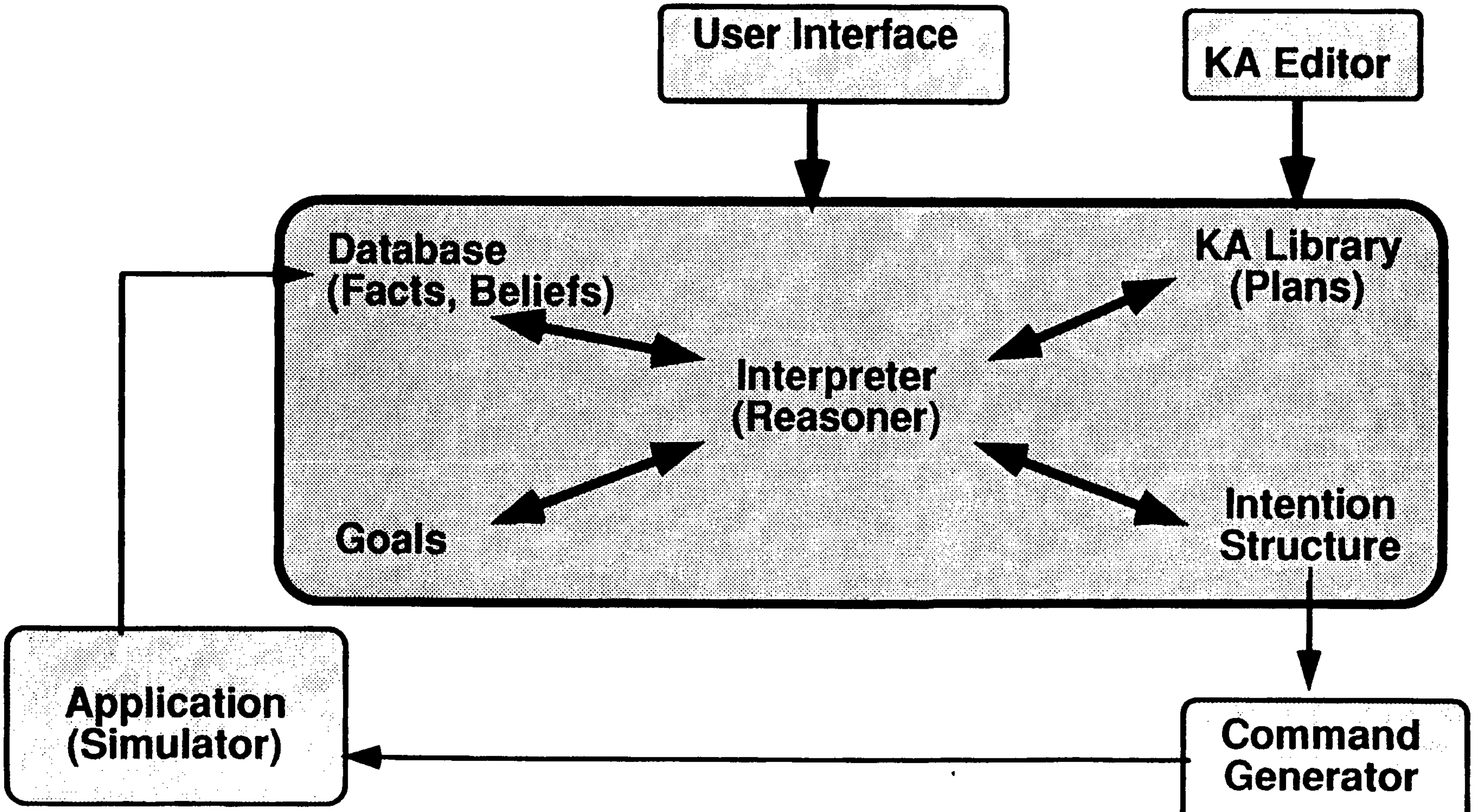
LESC/J. TRAN
EF5/H. HIERS

FEB. 7, 1990

ARCHITECTURE

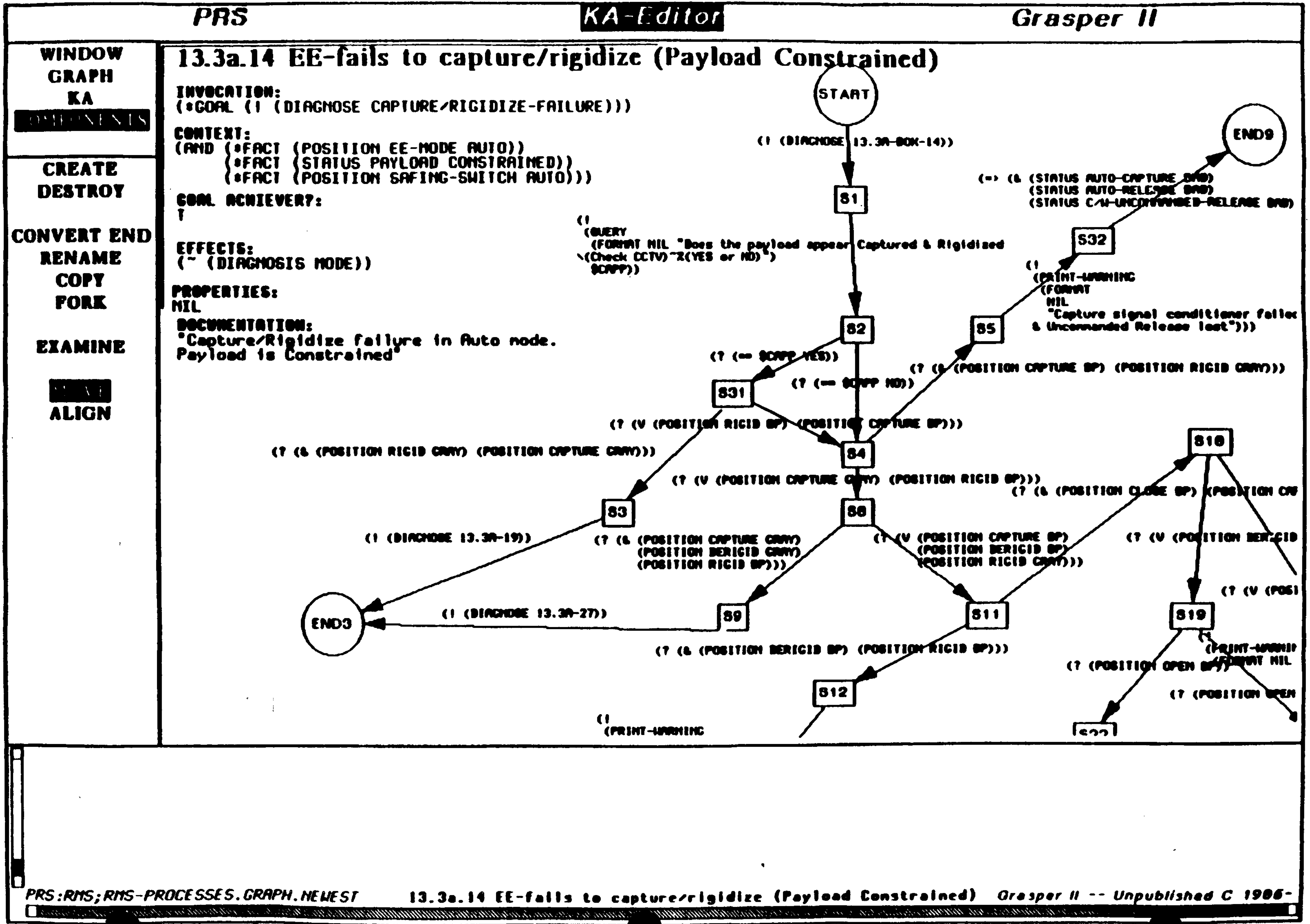


PRS ARCHITECTURE



PRS KA Editor

The PRS KA editor supports inputs of procedural information as a set of KAs. Each KA description includes a network of labeled nodes and edges, as well as an invocation condition that describes the situations in which that KA is applicable and useful.



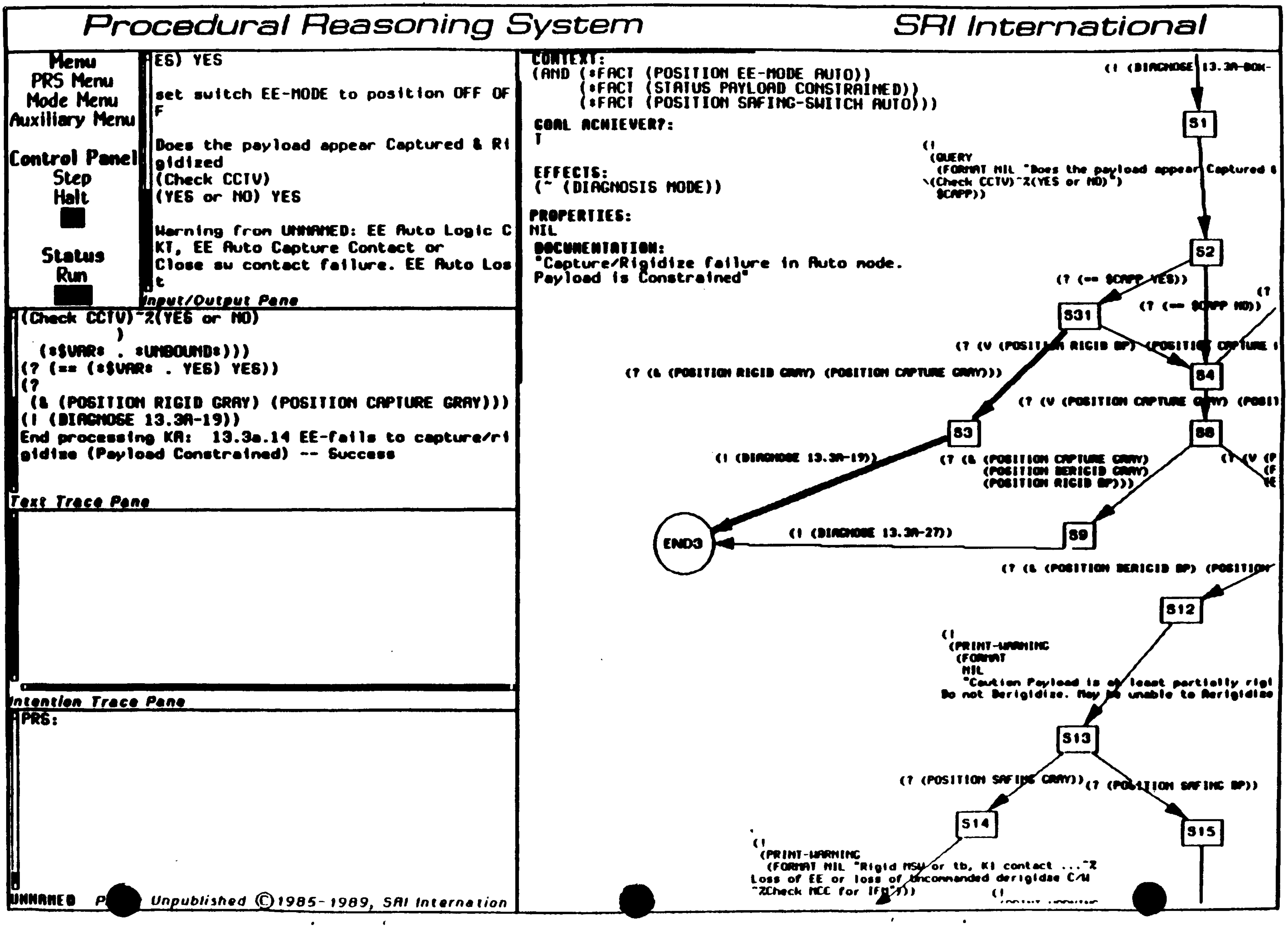
13.3a.14 EE-fails to capture/rigidize (Payload Constrained)

```
INVOCATION:
(*GOAL (! (DIAGNOSE CAPTURE/RIGIDIZE-FAILURE)))
CONTEXT:
(AND (*FACT (POSITION EE-MODE AUTO))
     (*FACT (STATUS PAYLOAD CONSTRAINED))
     (*FACT (POSITION SAFING-SWITCH AUTO)))
GOAL ACHIEVER?:
T

EFFECTS:
(~ (DIAGNOSIS MODE))
PROPERTIES:
NIL
DOCUMENTATION:
"Capture/Rigidize failure in Auto mode.
Payload is Constrained"
```

PRS:RMS;RMS-PROCESSES.GRAPH.NEWEST 13.3a.14 EE-fails to capture/rigidize (Payload Constrained) Grasper II -- Unpublished C 1986-

PRS Interface

The PRS interface is designed to manipulate data and to provide information about the various PRS agents (systems) the user has set up. The PRS interface can be used to create or delete PRS agents, load KA files, trace specific KAs, establish goals, query the database, and in general, perform the operations necessary to deal with the construction and operation of PRS agents.



```
ES) YES

set switch EE-MODE to position OFF OF
F

Does the payload appear Captured & Ri
gidized
(Check CCTV)
(YES or NO) YES

Warning from UNNAMED: EE Auto Logic C
KT, EE Auto Capture Contact or
Close sw contact failure. EE Auto Los
t
```

```
(Check CCTV)~%(YES or NO)
          )
   (*$VAR* . *UNBOUND*)))
(? (== (*$VAR* . YES) YES))
(?
  (& (POSITION RIGID GRAY) (POSITION CAPTURE GRAY)))
(! (DIAGNOSE 13.3A-19))
End processing KA:  13.3a.14 EE-fails to capture/ri
gidize (Payload Constrained) -- Success
```

```
PRS:
```

```
CONTEXT:
(AND (*FACT (POSITION EE-MODE AUTO))
     (*FACT (STATUS PAYLOAD CONSTRAINED))
     (*FACT (POSITION SAFING-SWITCH AUTO)))
GOAL ACHIEVER?:
T

EFFECTS:
(~ (DIAGNOSIS MODE))

PROPERTIES:
NIL
DOCUMENTATION:
"Capture/Rigidize failure in Auto mode.
Payload is Constrained"
```

```
(! (DIAGNOSE 13.3A-BOX-

(!
 (QUERY
  (FORMAT NIL "Does the payload appear Captured &
\(Check CCTV)~%(YES or NO)")
  $CAPP))

(? (== $CAPP YES))
(? (== $CAPP NO))
(?
(? (V (POSITION RIGID BP) (POSITION CAPTURE
(? (& (POSITION RIGID GRAY) (POSITION CAPTURE GRAY)))
(? (V (POSITION CAPTURE GRAY) (POSI
(! (DIAGNOSE 13.3A-19))
(? (& (POSITION CAPTURE GRAY)
      (POSITION DERIGID GRAY)
      (POSITION RIGID BP)))
(! (DIAGNOSE 13.3A-27))
(? (& (POSITION DERIGID BP) (POSITION

(!
 (PRINT-WARNING
  (FORMAT
   NIL
   "Caution Payload is at least partially rigi
Do not Derigidize. May be unable to Rerigidize

(? (POSITION SAFING GRAY))
(? (POSITION SAFING BP))

(!
 (PRINT-WARNING
  (FORMAT NIL "Rigid MSU or tb, Kl contact ...~%
Loss of EE or loss of Uncommanded derigidze C/W
~%Check MCC for IFM")))
(!
```

UNNAMED P Unpublished ©1985-1989, SRI Internation

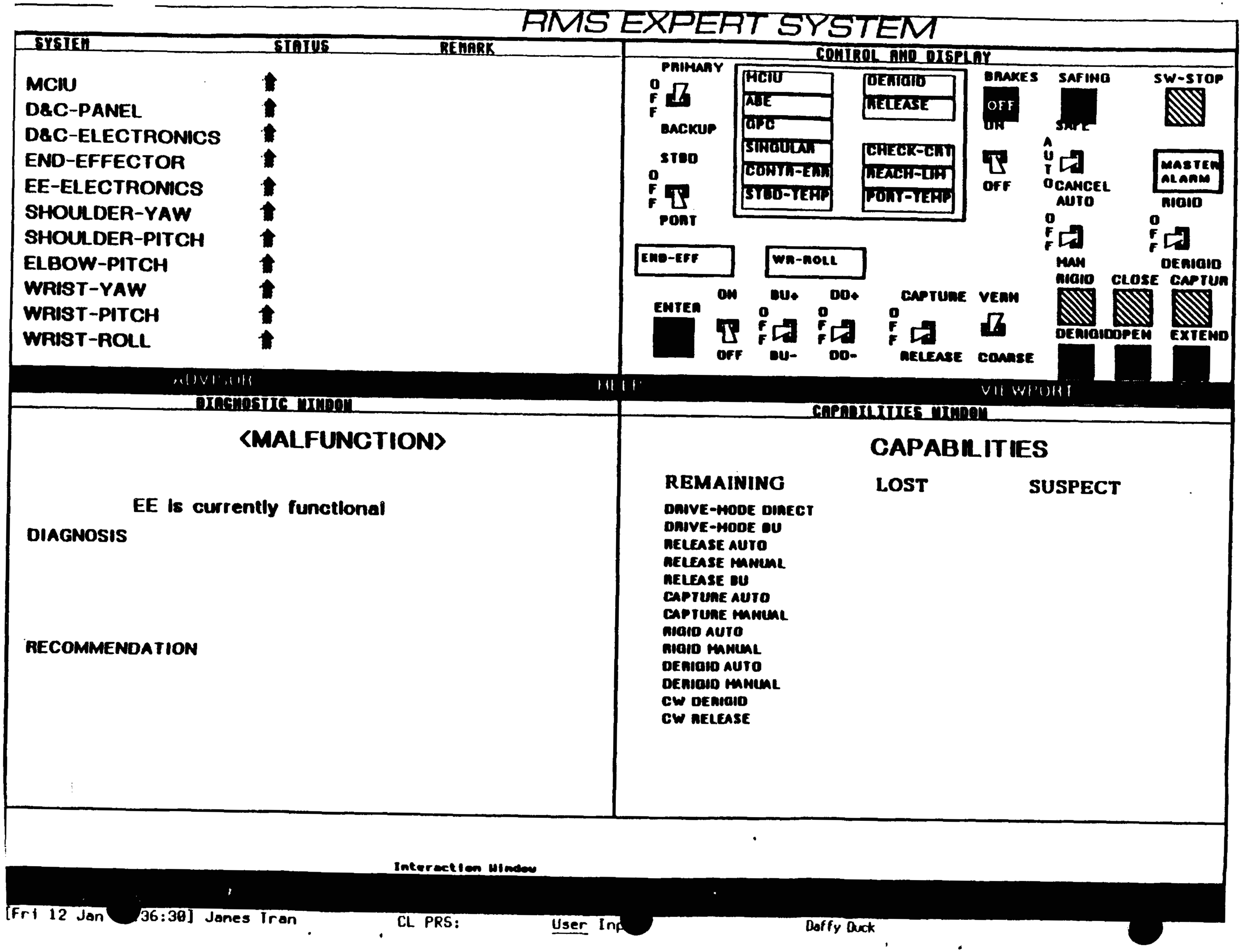
RMS Crew Interface - Initial State

The RMS Crew Interface is designed to present a simple and friendly environment for the crew to interact with the RMS expert system. It displays information vital to the orbiter RMS operation, and accepts commands from the RMS crew though mouse inputs. The initialized state of the interface shows 4 main windows and a menu pane in the middle.

The top left window provides a quick reference of the state of each component of the RMS, including a status arrow and a brief remark explaining its status. The right top window displays the mouse-sensitive RMS switches, talkbacks, and dials from the A8U and A8L panels, which allows the user to set switches or dials as prompted by the expert system. The center menu gives the user a means of controlling the interface and the PRS shell, including loading of the PRS database and initializing and running a procedure. The lower left and right windows display the results of the execution of RMS operational and malfunction procedures in format familiar to the RMS crew, e.g. type of malfunction, diagnosis, quick action, and status of the RMS capabilities.

When the RMS expert system has determined the diagnosis and recommendation of a malfunction, this information along with the status of the component's capabilities will be immediately displayed. All of this information will then be stored for future reference or until the component's status has changed.

If multiple malfunctions occur, the user can click the mouse on one of the problem components located in the status window to display its diagnosis information.



SYSTEM	STATUS	REMARK
MCIU	⬆	
D&C-PANEL	⬆	
D&C-ELECTRONICS	⬆	
END-EFFECTOR	⬆	
EE-ELECTRONICS	⬆	
SHOULDER-YAW	⬆	
SHOULDER-PITCH	⬆	
ELBOW-PITCH	⬆	
WRIST-YAW	⬆	
WRIST-PITCH	⬆	
WRIST-ROLL	⬆	

<MALFUNCTION>

EE is currently functional

DIAGNOSIS

RECOMMENDATION

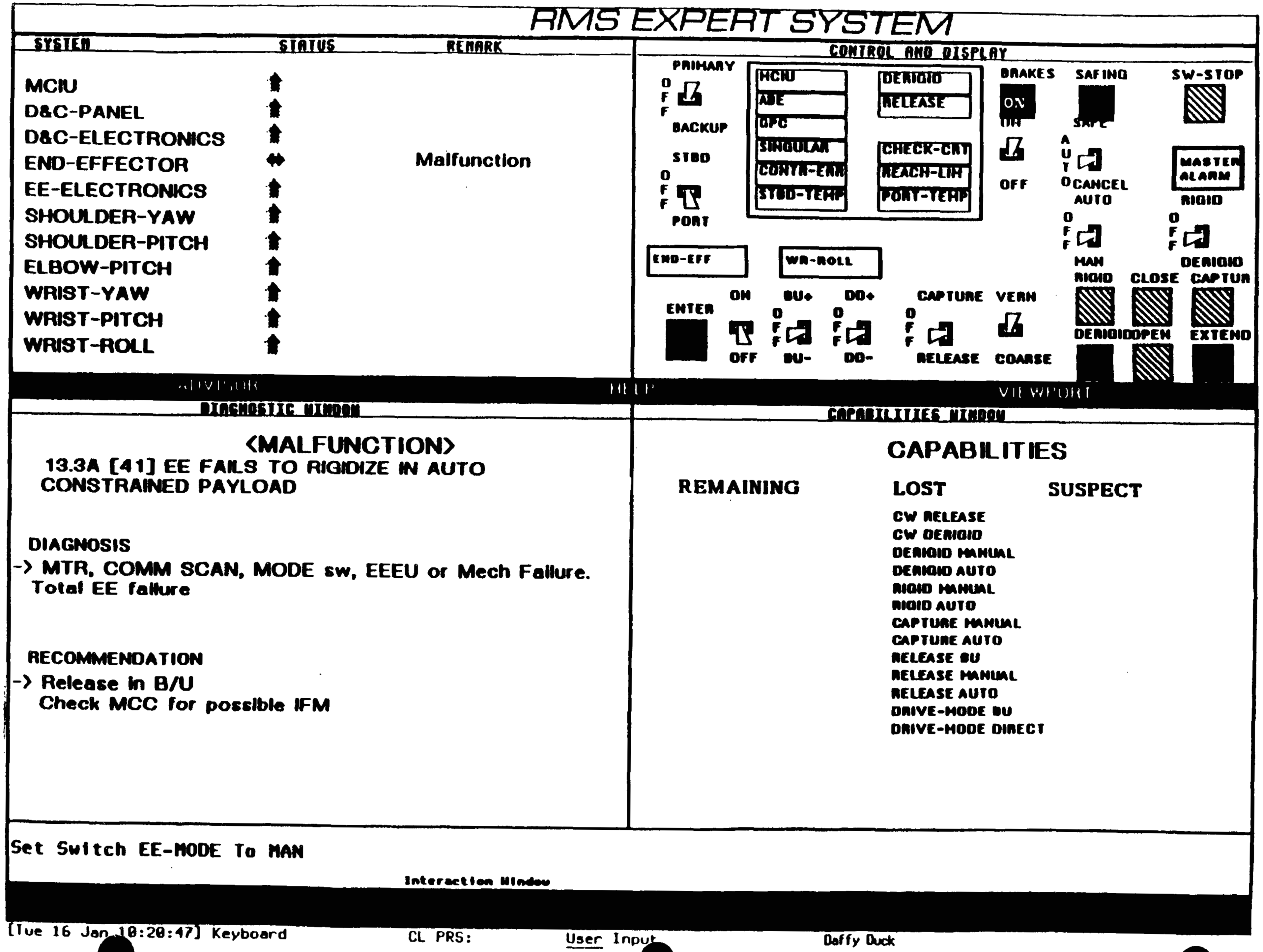
CAPABILITIES

REMAINING	LOST	SUSPECT
DRIVE-MODE DIRECT		
DRIVE-MODE BU		
RELEASE AUTO		
RELEASE MANUAL		
RELEASE BU		
CAPTURE AUTO		
CAPTURE MANUAL		
RIGID AUTO		
RIGID MANUAL		
DERIGID AUTO		
DERIGID MANUAL		
CW DERIGID		
CW RELEASE		

[Fri 12 Jan 36:30] Janes Tran CL PRS: User Inp Daffy Duck

RMS Crew interface - final state

This slide shows the RMS Crew Interface after a RMS malfunction has been successfully diagnosed by the expert system. The malfunction, diagnosis, and recommendation has been displayed and the status of the End Effector has become questionable. The remark notifies the user that the manual mode is bad. The simulated display and control panel shows the current setting reflecting the current state of the database. The RMS capabilities that have been lost due to the malfunction as well as the capabilities that still remain are listed in the lower right window. The lower left window displays the malfunction nature, the relevant diagnosis and the recommendation for actions.



SYSTEM	STATUS	REMARK
MCIU	↑	
D&C-PANEL	↑	
D&C-ELECTRONICS	↑	
END-EFFECTOR	↔	Malfunction
EE-ELECTRONICS	↑	
SHOULDER-YAW	↑	
SHOULDER-PITCH	↑	
ELBOW-PITCH	↑	
WRIST-YAW	↑	
WRIST-PITCH	↑	
WRIST-ROLL	↑	

DIAGNOSTIC WINDOW

<MALFUNCTION>

13.3A [41] EE FAILS TO RIGIDIZE IN AUTO
CONSTRAINED PAYLOAD

DIAGNOSIS

-> MTR, COMM SCAN, MODE sw, EEEU or Mech Failure.
Total EE failure

RECOMMENDATION

-> Release in B/U
Check MCC for possible IFM

CAPABILITIES WINDOW

CAPABILITIES

REMAINING	LOST	SUSPECT
	CW RELEASE	
	CW DERIGID	
	DERIGID MANUAL	
	DERIGID AUTO	
	RIGID MANUAL	
	RIGID AUTO	
	CAPTURE MANUAL	
	CAPTURE AUTO	
	RELEASE BU	
	RELEASE MANUAL	
	RELEASE AUTO	
	DRIVE-MODE BU	
	DRIVE-MODE DIRECT	

Set Switch EE-MODE To MAN

Interaction Window

[Tue 16 Jan 10:20:47] Keyboard CL PRS: User Input Daffy Duck

RMS AUTOMATION POTENTIAL

	RMS ASTRONAUT	RMS AUTOMATION
Data Monitoring	YES	YES
Fault Detection	YES	YES
Select Applicable Procedures	YES	YES
Execute Procedures	YES	YES
Malfunction Diagnosis	YES	YES
Keep Track of Reasoning Path	YES	YES
Visual Contact	YES	NO
Ultimate Control	YES	NO
Consistency	NA	YES
Less Expensive	NA	YES
Require Flight Data File	YES	NO

FUTURE WORK

- **INTEGRATION WITH JSC/SYSTEMS ENGINEERING SIMULATOR FOR REAL-TIME EVALUATION**
- **EXPAND SYSTEM CAPACITY TO COVER OTHER SUB-COMPONENTS OF THE RMS**
- **DEVELOP SPECIFICATIONS FOR SSE ADA PROCEDURAL REASONING TOOL**

ADVANTAGES

- Reduce crew fatigue by advancing the decision making process to a higher level
- Result in a quicker response to an immediate condition
- Still allow for detailed analysis if necessary
- Provide real-time simulations
- Provide real-time modifications
- Provide real-time validations
- Compile a precise database of failures encountered and procedures taken in real-time
- Benefit ground crew as well in developing, verifying, and training with the procedures
- Facilitate multiple-failure analysis
- Reduce Flight Data File hardcopies

CONCERNS

- CPU time may not be available
- Expert system needs to be redundant in performance
- Manual override must be incorporated
- Expert system self-check may be required
- Expert system must be as least fail-safed by design

THERMAL EXPERT SYSTEM - TEXSYS DEVELOPMENT AND TEST REVIEW

SSF Evolution Conference

Houston, TX

February 6-8, 1990

The Thermal Expert System (TEXSYS) was initiated in 1986 as a cooperative project between ARC and JSC as a way to leverage on-going work at both centers. JSC contributed Thermal Control System (TCS) hardware and control software, TCS operational expertise, and integration expertise. ARC contributed expert system and display expertise. The first years of the project were dedicated to parallel development of expert system tools, displays, interface software and TCS technology and procedures by a total of four organizations (two at ARC, two at JSC). A demonstration was planned as the final project milestone.

BACKGROUND

JSC DEVELOPING STATION THERMAL CONTROL SYSTEM

- New two-phase (liquid/vapor) technology
- Operational expertise

ARC CONDUCTING SYSTEMS AUTONOMY DEMONSTRATION PROGRAM

- Development of expert system and display tools
- Goal of real-time control and FDIR of a system

COOPERATIVE PROJECT

THERMAL EXPERT SYSTEM SELECTED 1986

- Parallel development of expert system tools, thermal technology, interface software
- Combined effort of two ARC (FL & RIS) and two JSC (EC & EF) organizations
- Demonstration planned as final milestone

TEXSYS is one of the first real time expert systems to perform control on a large, complex physical system. It was actually developed in an iterative fashion, with its first step to interact with a smaller TCS brassboard test article. The system was then upgraded to handle the actual test article and more faults, and was progressively tested and corrected to its final demonstration configuration. It uses model-based reasoning (327 rules and 3,493 frames) and its networking of software interfaces must fit into a 15 second cycle time.

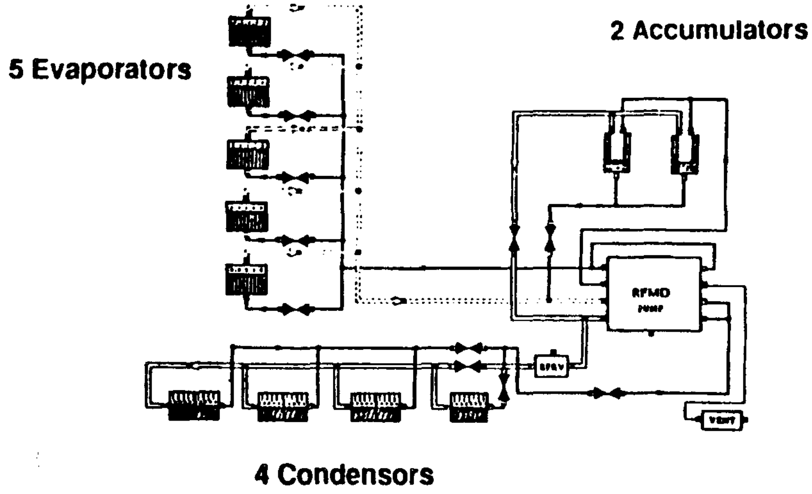
SYSTEMS AUTONOMY DEMONSTRATION PROJECT

ADVANCED AUTOMATION DEMONSTRATION OF SPACE STATION FREEDOM THERMAL CONTROL SYSTEM

TECHNOLOGY CHALLENGE

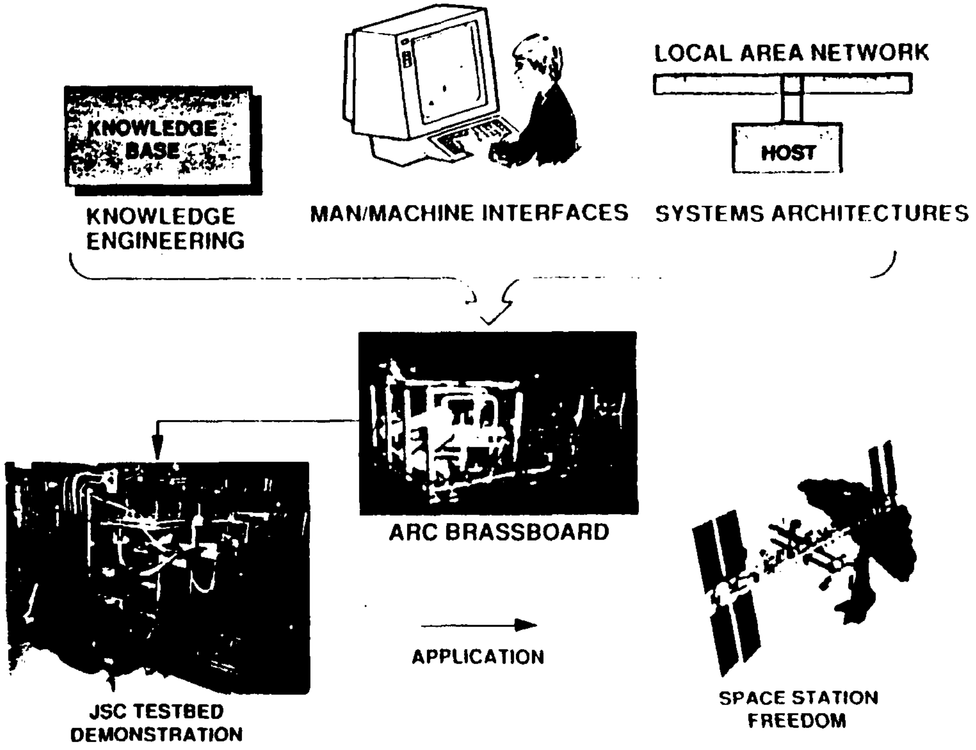
EXPERT SYSTEM REALTIME CONTROL OF A COMPLEX ELECTRO-MECHANICAL SYSTEM
- ***Advanced Thermal Technology***
- ***Complex Physical System***

THERMAL CONTROL SYSTEM (TCS)
Two-Phase Anhydrous Ammonia System



TECHNOLOGY IMPLEMENTATION

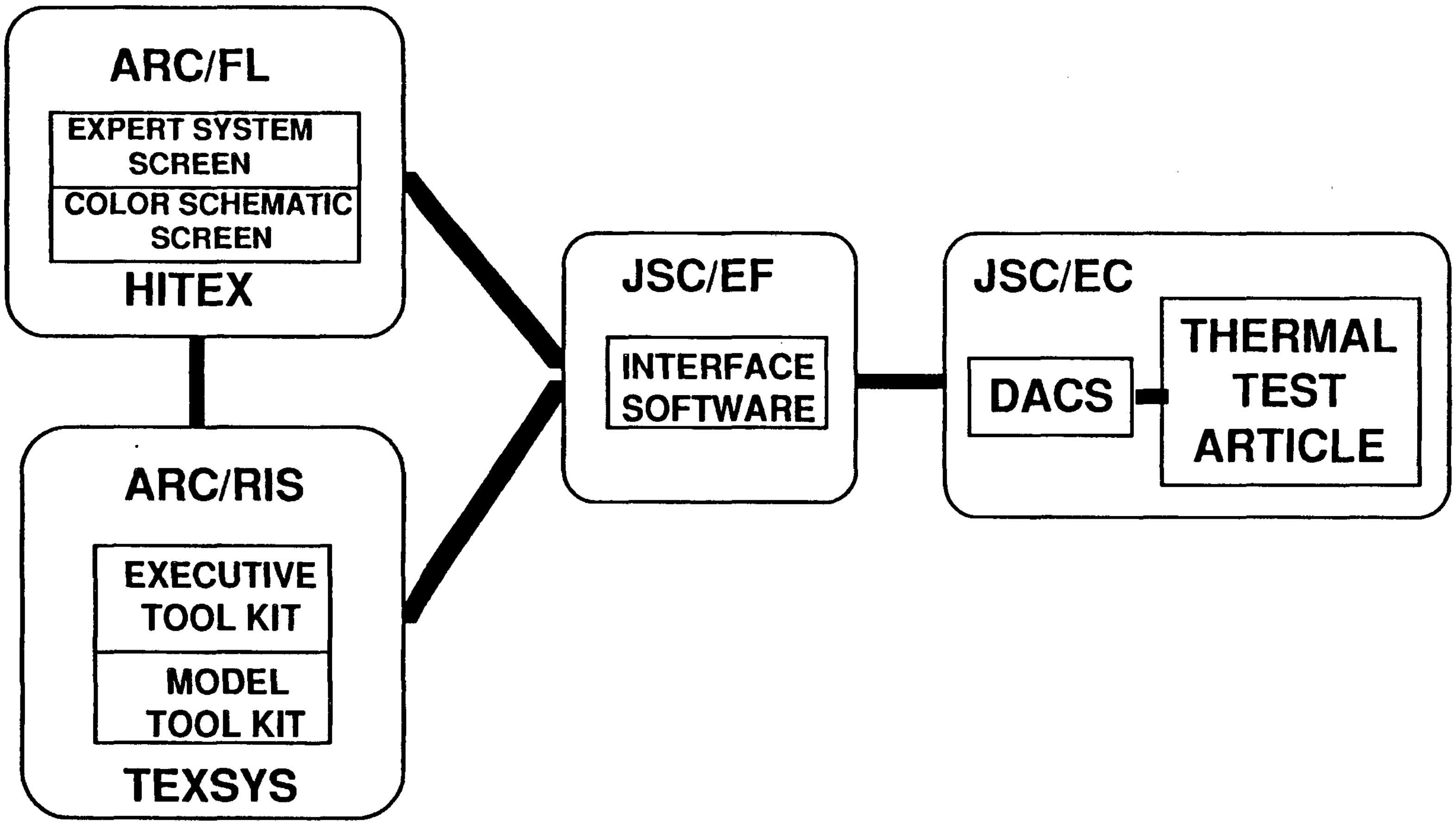
JOINT ARC/JSC DEMONSTRATION



NASA AMES RESEARCH CENTER
INFORMATION SCIENCES DIVISION

TEXSYS consisted of four major software units layered on top of one another. JSC developed both the conventional control software that interacts with the test article and its interface software to the expert system. ARC developed the Thermal Expert System (TEXSYS) and the human interface to TEXSYS (HITEX). TEXSYS and HITEX each ran on a dedicated Symbolics computer, while the conventional control software ran on two microVax computers. All the computers were networked to one another, with the interface software distributed between all the computers.

OVERVIEW OF TEXSYS



ARC/FL

EXPERT SYSTEM SCREEN

COLOR SCHEMATIC SCREEN

HITEX

ARC/RIS

EXECUTIVE TOOL KIT

MODEL TOOL KIT

TEXSYS

JSC/EF

INTERFACE SOFTWARE

JSC/EC

DACS

THERMAL TEST ARTICLE

TEXSYS was designed to conduct both real time control and fault detection, isolation and recovery (FDIR) of the thermal test article. From a list of 38 potential faults, ten faults were selected for implementation and demonstration in TEXSYS. The test article was configured to allow detection of all 10 faults with varying levels of automatic recovery.

SPECIFIC FUNCTIONALITY TO BE DEMONSTRATED

REAL-TIME CONTROL

STARTUP

NORMAL OPERATIONS

SHUTDOWN

FAULT DETECTION, ISOLATION, AND RECOVERY OF 10 COMPONENT LEVEL FAULTS

1. Slow Leak
2. Pump Motor Failure
3. Single Evaporator Blockage
4. High Coolant Sink Temperature
5. Temp Valve Failure
6. NCG Buildup
7. Temp Valve Actuator Failure
8. Excessive Heat Load on Single Evaporator
9. Accumulator Position Sensor Failure
10. Pressure Sensor Failure

The TEXSYS project culminated with 5 months of integration and checkout, followed by a one week demonstration. TEXSYS successfully conducted all of its control and FDIR procedures. It proved to be generally reliable for conducting fault detection. Both the fault detection capability and the graphical displays were significant improvements over the conventional controller. Slowdowns in processing time decreased the reliability of the expert system. Future upgrades to the system should address the slowdowns and improve the fault detection explanation capability.

RESULTS

SOFTWARE INTEGRATION/CHECKOUT PERFORMED AT JSC MARCH - AUGUST 1989

- Simple interface tests approx 3 weeks
- Playback of pre-recorded test article data approx 3 months
- Actual interaction with live test article approx 6 weeks

DEMONSTRATION WEEK (8/28 - 9/1/89) SUCCESSFULLY SHOWED ALL NORMAL OPERATING PROCEDURES AND FAULT DETECTION ON ALL 10 FAULTS

STRENGTHS

- Significant improvement over previous capability
- Excellent graphical displays
- Generally reliable Fault Detection capability

WEAKNESSES

- Slowdowns in processing time decreased reliability, ease of use
- On-screen explainations need to be enhanced

Advanced automation technology provides useful tools to engineers attempting to capture and utilize design and operational expertise. TCS engineers can use this technology to better design thermal systems for future programs.

One of the biggest difficulties has been, and continues to be in the ability to design a system and in parallel design and codify its operational procedures. Advanced automation tools are beginning to add extra flexibility over conventional tools to better allow the capture of design and operational expertise as a system develops. Further research is required to find effective tools to checkout and certify this type of software.

The presentation concludes with self-descriptive two page list of Lessons Learned that were gained during the TEXSYS development and test.

CONCLUSIONS

1. TCS Engineers better prepared to develop automation software for Space Station, Advanced Programs.
2. Expert System community has more experience with large model-based expert systems for real-time process control.
3. Codifing new hardware operating procedures using new advanced automation techniques is a challenge.
4. Further research is needed into use of simulation software and other tools to develop and checkout expert systems.

LESSONS LEARNED

1. Identify user, focus on his application. Application and knowledge engineers should work together to:
   - Develop requirements early in the project
   - Define the operating and fault diagnosis procedures
   - Conduct a code walkthrough
   - Conduct hardware/software testing

2. New technology adds development time.
   - Application operational immaturity required extra time to develop fault diagnosis and recovery procedures
   - Real-time model-based expert system tools required development and checkout time

3. Iterative coding and testing is an effective expert system development process.
   - Brassboard testing stressed performance
   - Playback of pre-recorded test article data improved accuracy
   - Full-up testing is a final step

LESSONS LEARNED

4. Slow system, dedicated computers ease real-time performance problems.
   - TCS parameters, in general, change slowly with time (~seconds)
   - TEXSYS project employed two Symbolics computers (TEXSYS and HITEX) and two microVax computers (Conventional control and Interface software)
   - Network and microVaxes were tuned to optimize performance

5. Clean interfaces eased integration between conventional and expert system code.
   - ICD
   - Modular subroutines in conventional software

SUMMARY OF ASTRONAUTS' INPUTS CONCERNING AUTOMATION

AN ASSESSMENT OF THE POTENTIAL FOR INCREASED PRODUCTIVITY

FEBRUARY 7, 1990

DAVE WEEKS

NASA/MARSHALL SPACE FLIGHT CENTER

NASA

ASTRONAUT INPUTS ON A&R

BACKGROUND:

ADVANCED DEVELOPMENT PROGRAM STAFF AT HQ RECOGNIZED A NEED TO ENSURE THAT PROGRAM ACTIVITIES STRONGLY CORRELATED WITH SPACE STATION ONBOARD & GROUND SUPPORT USER COMMUNITY NEEDS AS THE SPACE STATION EVOLVES

OBJECTIVE:

CALIBRATE ADVANCED DEVELOPMENT ACTIVITIES FOR THE SPACE STATE BY -

- REVIEW LESSONS LEARNED FROM PREVIOUS MANNED SPACE EXPERIENCES & ANALOGS
- GAIN FURTHER INSIGHT FROM SPACE STATION USER COMMUNITY ON NEEDS/DESIRES
- STRENGTHEN A&R ADVOCACY WITHIN THE ASTRONAUT OFFICE

ASTRONAUT INPUTS ON A&R

APPROACH:

PERFORM STUDY WHICH EMPHASIZED INCREASED PRODUCTIVITY BY FOCUSING ON -

- LESSONS LEARNED FROM RELATED EXPERIENCES
  - SKYLAB
  - SPACELAB
  - OTHER STS
  - SOVIET SPACE STATION
  - U.S. NUCLEAR SUBMARINE PROGRAM
  - ANTARCTIC RESEARCH STATIONS
- CREW TIME REQUIREMENTS
- FLIGHT CREW & GROUND SUPPORT PERCEPTIONS
  - INTERVIEWED 23 CURRENT/FORMER ASTRONAUTS & PAYLOAD SPECIALISTS
  - INTERVIEWED 22 GROUND SUPPORT PERSONNEL
  - SURVEYED 32 OF ASTRONAUT/PAYLOAD SPECIALIST COMMUNITY

ASTRONAUT INPUTS ON A&R

ASTRONAUTS/PAYLOAD SPECIALISTS INTERVIEWED

Astronauts/Payload Specialists	Missions Flown
John-David Bartoe	STS 51-F (Spacelab 2)
Gerald P. Carr	Skylab 4
N. Jan Davis	assigned to STS 47 (Spacelab J)
Bonnie Dunbar	STS 61-A, STS 32
Owen K. Garriott	Skylab 3, STS 9 (Spacelab 1)
Edward G. Gibson	Skylab 4
Greg Harbaugh	assigned to STS 39
Henry w. Hartsfield	STS 4, STS 41-D, STS 61-A
David C. Hilmers	STS 51-J, STS 26
Jeffrey A. Hoffman	STS 51-D
Joseph P. Kerwin	Skylab 2
Byron K. Lichtenberg	STS 9 (Spacelab 1)
John M. Lounge	STS 51-I, STS 26
Jack R. Lousma	Skylab 3, STS 3
Story F. Musgrave	STS 6, STS 51-F (Spacelab 2), STS 33
Claude Nicollier	assigned to STS 46
Robert F. Overmeyer	STS 5, STS 51-B (Spacelab 3)
Robert A. Parker	STS 9 (Spacelab 1)
William R. Pogue	Skylab 4
Jerry L. Ross	STS 61-B, STS 27
Rhea m. Seddon	STS 51-D
Robert Springer	STS 29
John W. Young	Gemini III, Gemini IX, Apollo 10, Apollo 16, STS 1, STS 9 (Spacelab 1)

ASTRONAUT INPUTS ON A&R

RESULTS

A final report will be out this month entitled:

SPACE STATION FREEDOM AUTOMATION AND ROBOTICS:

AN ASSESSMENT OF THE POTENTIAL FOR INCREASING PRODUCTIVITY

HIGHLIGHTS OF THE ASTRONAUT/PAYLOAD SPECIALIST SURVEY BY QUESTIONNAIRE:

Philosophically, 81% favor using advanced automation to increase Space Station productivity

Safetywise, 93% rated FDIR and 84 % rated Exception Reporting/Filtering having potential to contribute some increase to significant improvements

The questionnaire evoked 611 responses regarding 26 specific A&R applications
- 76% indicated having potential for at least some increase in productivity
- 7% seen with potential for some productivity decrease or significant problems
- 17% viewed having negligible impact on productivity

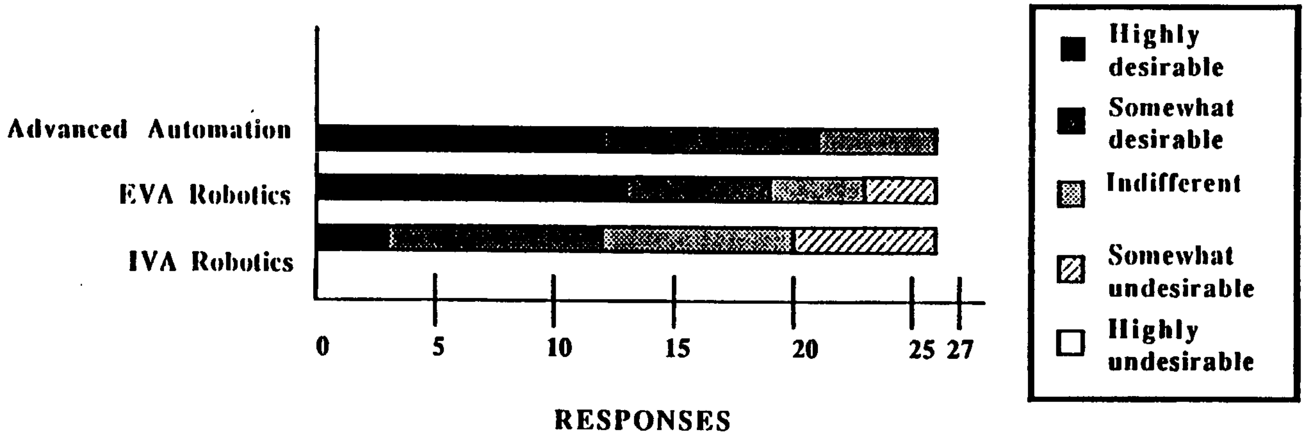
ASTRONAUT INPUTS ON A&R

ASTRONAUT COMMUNITY INPUTS REGARDING ADVANCED AUTOMATION

- SIMPLE, STANDARDIZED HUMAN INTERFACE (IDIOT-PROOF)
- PROVIDE FLEXIBLE OPERATIONS CAPABILITY
- USER (VERSUS TECHNOLOGY/DEVELOPER) ORIENTED
- DEVELOP & IMPLEMENT THE EASIER APPLICATIONS FIRST
- HELP THE USER TO DO THE JOB EASIER INSTEAD OF MORE DIFFICULT
- INCLUDE "WHAT-IF?" CAPABILITY
- BACKUP MODE OF OPERATION
- SYSTEM MUST BE ABLE TO EXPLAIN CONCLUSIONS/ACTIONS
- AUTOMATE:

  TEDIOUS, REPETITIVE TASKS
  TIME-DEPENDENT TASKS
  CALIBRATION & ALIGNMENT TASKS
  ROBOTIC SET-UP FOR EVA (FOOT RESTRAINTS)

ASTRONAUT INPUTS ON A&R



ASTRONAUT PHILOSOPHICAL VIEWS REGARDING A&R

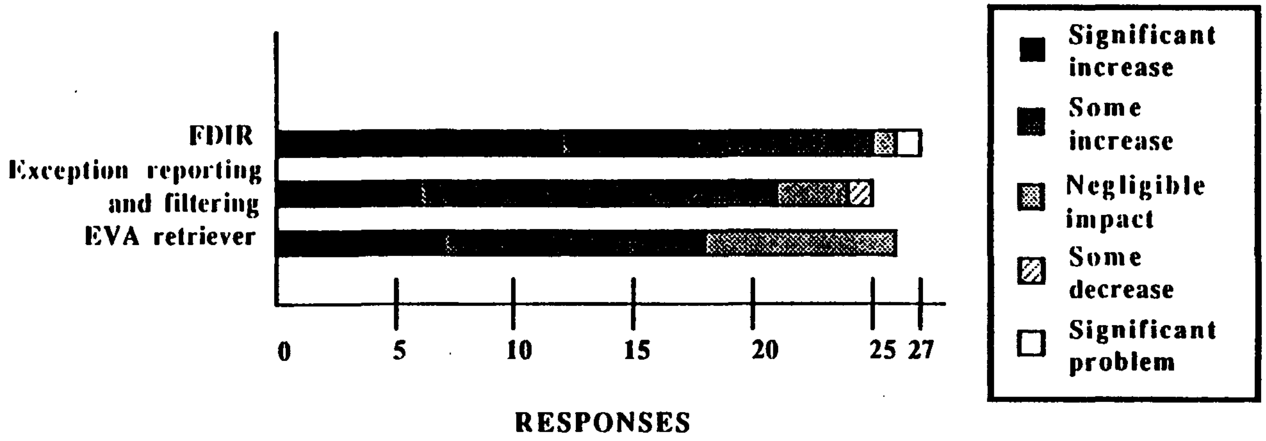
ASTRONAUT INPUTS ON A&R

ASTRONAUT SUPPORTED AUTOMATION APPLICATIONS FOR IMPROVING PRODUCTIVITY

OF 32 CURRENT/FORMER ASTRONAUTS & PAYLOAD SPECIALISTS SURVEYED:

- AUTOMATED RECORD KEEPING & DOCUMENTATION (100%)
- AUTOMATED INVENTORY MANAGEMENT (96%)
- AUTOMATED FDIR (93%)
- IMPROVED HUMAN-COMPUTER INTERFACES (92%)
- ROBOTIC CONSTRUCTION (92%)
- ROBOTIC INSPECTION (88%)
- EXCEPTION REPORTING/ALARM FILTERING (88%)
- EXTERNAL CAMERA/LIGHT POINTING (87%)
- ROBOTIC EXTERNAL REPAIRS (85%)
- AUTOMATED TRENDS ANALYSES (INCIPIENT FAILURE DETECTION) (85%)
- CHECKLIST AUTOMATION (85%)

ASTRONAUT INPUTS ON A&R



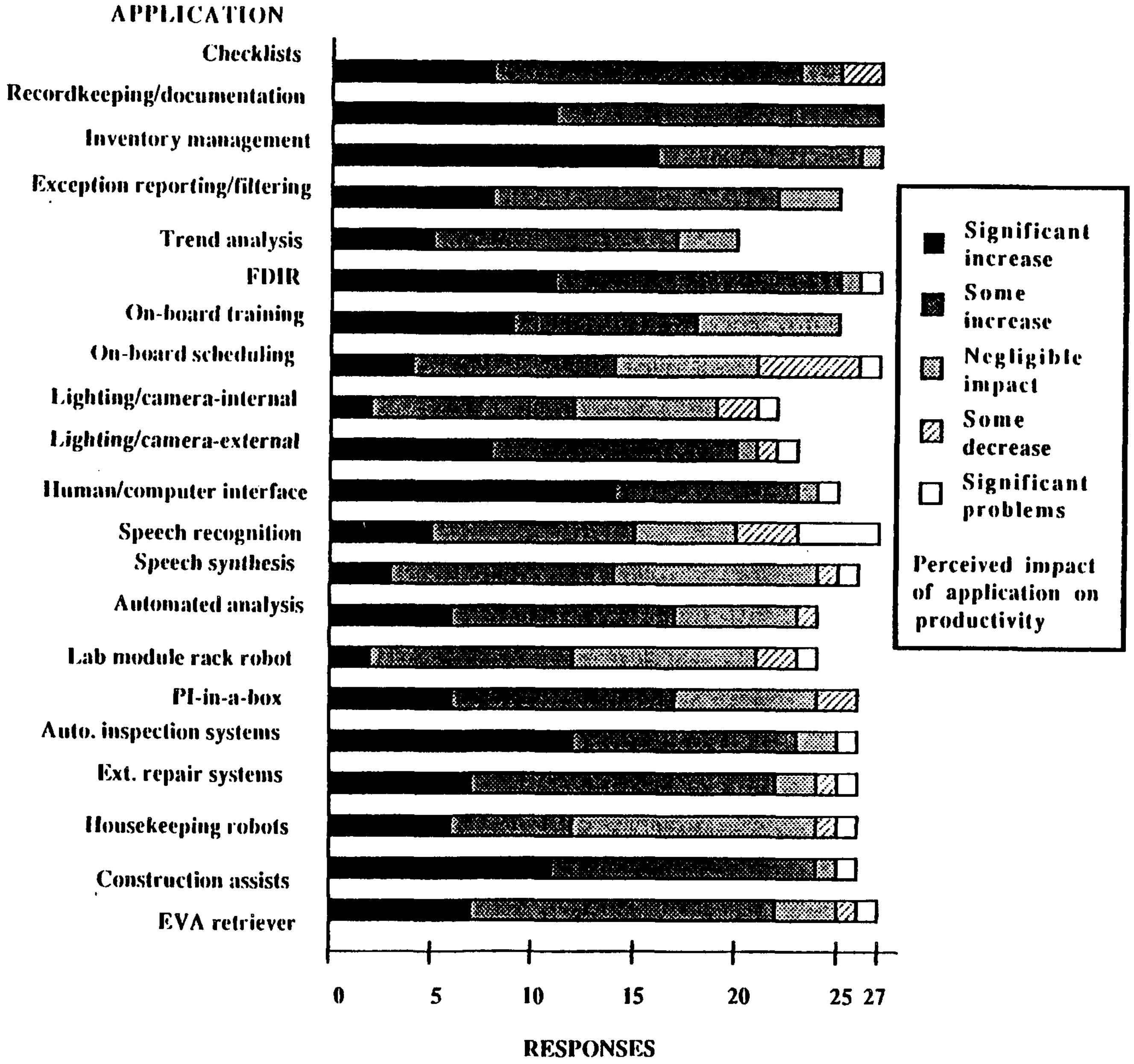
ASTRONAUT RATINGS OF SAFETY IMPACTS OF A&R APPLICATIONS

ASTRONAUT INPUTS ON A&R

ASTRONAUT SUPPORTED AUTOMATION APPLICATIONS (CONTINUED)

- SYSTEMS MONITORING & CONTROL (82%)
- EVA RETRIEVER ROBOTICS (81%)
- PAYLOAD-SPECIFIC AUTOMATION (79%)
- ON-BOARD TRAINING SYSTEMS (72%)
- PAYLOAD AUTOMATED DATA ANALYSIS (71%)
- PRINCIPAL INVESTIGATOR IN A BOX TYPE SYSTEMS (65%)
- INTERNAL CAMERA/LIGHTING POINTING (58%)
- SPEECH RECOGNITION (56%)
- SPEECH SYNTHESIS (54%)
- ON-BOARD SCHEDULING/RESCHEDULING CAPABILITY (52%)
- IVA RACK ROBOT (50%)
- IVA HOUSEKEEPING ROBOTICS (46%)

ASTRONAUT INPUTS ON A&R



ASTRONAUT ESTIMATES OF PRODUCTIVITY IMPACT OF A&R APPLICATIONS

ASTRONAUT INPUTS ON A&R

CONCLUSIONS

- STRONG A&R SUPPORT FROM THE ASTRONAUT COMMUNITY FOR APPROPRIATE APPLICATIONS
  - SKYLAB VETERANS MORE SUPPORTIVE OF A&R THAN STS VETERANS
  - STRONGLY SUPPORTED APPLICATIONS INCLUDE
    - INVENTORY MANAGEMENT
    - IMPROVED HUMAN-COMPUTER INTERFACES
    - INSPECTION WITH EVA TELEROBOTICS

- AREAS OF GREATEST POTENTIAL FOR PRODUCTIVITY IMPACT ON SPACE STATION INCLUDE:
  - PAYLOAD OPERATIONS
  - INVENTORY MANAGEMENT
  - SYSTEM MONITORING & CONTROL (INCLUDING FDIR WITH DIAGNOSIS & INTELLIGENT RECOVERY)

- PERSONAL PRODUCTIVITY FALLS OFF SHARPLY AT 8 CONT. HOURS; SPACELAB PRODUCTIVE TIME 70%

- ADDITIONAL PRODUCTIVITY IMPROVEMENT POTENTIAL EXISTS FOR GROUND-BASED SUPPORT:
  - SPACE STATION CONTROL CENTER FUNCTIONS
  - OPERATIONS PLANNING AND INTEGRATION ACTIVITIES
  - TRAINING
  - SOFTWARE MAINTENANCE

- EVA ROBOTICS ALSO HAS GREAT POTENTIAL:
  - MOST COST EFFECTIVE (& SIMPLEST) TO DECREASE ASTRONAUT EVA TIME & INCREASE PRODUCTIVITY - TRANSFER CONTROL OF ROBOTIC ELEMENTS TO GROUND FOR SELECTED ASSY/MAINT/INSPECTION TASKS

- BETTER DEFINITION OF SSF ACTIVITIES/CREW TASKS NEEDED TO PROVIDE FIRM QUANT. EST. OF BENEFITS

ASTRONAUT INPUTS ON A&R

RECOMMENDATIONS

- DEVELOPMENT OF ADVANCED AUTOMATION & ROBOTICS BE ACTIVELY PURSUED; EMPHASIZED AREAS INCLUDE KBS FOR FLIGHT/GROUND SPT SYSTEMS, HUMAN-SYSTEM INTERFACES, & EVA TELEROBOTICS
- FOCUS ON NEAR-TERM TECHNICAL FEASIBILITY, HIGH POTENTIAL FOR SAVING CREW TIME & REDUCING GROUND SUPPORT STAFFING, AND ACCEPTANCE/SUPPORT BY USERS. INCREASE EMPHASIS ON INVENTORY MANAGEMENT
- ENSURE ADEQUATE PROVISIONS MADE IN SYSTEM DESIGN TO ACCOMMODATE FUTURE INTRODUCTION OF A&R TECHNOLOGY
- DEVELOP ADDITIONAL DATA FOR MORE PRECISE QUANTITATIVE ESTIMATES OF IMPACT OF SPECIFIC SYSTEMS ON PRODUCTIVITY & LIFE-CYCLE COST; INCLUDE COLLECTION OF WORKLOAD & ACTIVITY DURATION DATA FROM SSF ONCE PERMANENTLY MANNED
- PERFORM SYSTEMS ENGINEERING STUDY APPROACH TO TRADE ISSUES INVOLVING ALLOCATION OF FUNCTIONS TO A PERSON, MACHINE, OR COMBINATION THEREOF; TOP DOWN APPROACH SHOULD CONSIDER CREW ACTIVITIES IN TWO CATEGORIES:

  - OPERATIONS - ROUTINE DAILY EVENTS MIGHT BE REDUCED FROM 3 TO 2 HOURS/CREW MEMBER
  - MISSION ACTIVITIES - INVOLVING CREW EXPERIMENTS & NEW CREW JOBS WHICH HAS GREATER POTENTIAL FOR PRODUCTIVITY GAINS.

  THEN, FACTOR IN RELIABILITY, SAFETY, ETC., TO INDICATE HIGH PAYOFF APPLICATIONS

SESSION VI
GROUND OPERATIONS AUTOMATION

Session Chair:
Mr. John Muratore
NASA Lyndon B. Johnson Space Center

REAL-TIME DATA SYSTEM

MISSION OPERATIONS DIRECTORATE RECONFIGURATION MANAGEMENT OFFICE

SSF EVOLUTION SYMPOSIUM - GROUND SYSTEMS AUTOMATION
FEBRUARY 7, 1990

JOHN F. MURATORE

SSF EVOLUTION SYMPOSIUM GROUND SYSTEMS AUTOMATION

- GROUND SYSTEMS WILL BE CRITICAL IN EARLY SSF PROGRAM (1995-1997) DUE TO EXTENDED ASSEMBLY AND MAN-TENDED PHASES.
- TECHNOLOGY FOR INITIAL SSF GROUND SYSTEMS IS WELL-UNDERSTOOD AND EXCEPT FOR HIGH RATE FRONT END PROCESSING IS PRIMARILY NOT A TECHNICAL CHALLENGE
- DUE TO EVOLVING CONFIGURATION, INITIAL AUTOMATION WILL BE MODEST.
- AUTOMATION IS CRITICAL TO LONG-TERM COST-EFFECTIVENESS OF GROUND OPERATIONS.
- SEVERAL EMERGING TECHNOLOGIES HAVE LARGE POTENTIAL BENEFIT TO GROUND OPERATIONS.
- CHALLENGE IN GROUND SYSTEMS DESIGN IS ARCHITECTURE DESIGNED TO ACCEPT EMERGING TECHNOLOGIES.

EMERGING TECHNOLOGIES CAN POTENTIALLY BENEFIT GROUND OPERATIONS

- KNOWLEDGE BASED SYSTEMS - (RULE AND MODEL BASED)
- NETWORKING STANDARDS (ISO)
- WINDOWING AND GRAPHICS STANDARDS
- NEUTRAL NETWORKS
- REMOTE TELEROBOTIC OPERATIONS
- PARALLEL PROCESSING
- HIGH DEFINITION TV
- IMPROVED HUMAN/COMPUTER INTERFACES
- COMPUTER AIDED SOFTWARE ENGINEERING/SOFTWARE SUPPORT ENVIRONMENTS
- OPTICAL DISK
- DIGITAL VIDEO
- HYPERMEDIA
- NEW CPU TECHNOLOGIES
- ADA

CHALLENGE IS TO STRUCTURE GROUND SYSTEMS
TO ACCEPT EMERGING TECHNOLOGIES

- RTDS IS EXAMPLE MODEL - SHADOW FORCE
  - INTRODUCE NEW TECHNOLOGY IN PARALLEL IN OPERATIONAL LOCATION.
  - EVALUATE BY COMPARISON.
  - BACK OUT OLD TECHNOLOGY AS CONFIDENCE ESTABLISHED.
  - SHADOW APPROACH.

- T-FCR IS ALTERNATIVE MODEL - ALL UP APPROACH
  - BUILD ENTIRE CONTROL CENTER AND OPERATE IN PARALLEL.

Figure 7.- Console of the Integrated Communications Officer in the Mission Contr Center.

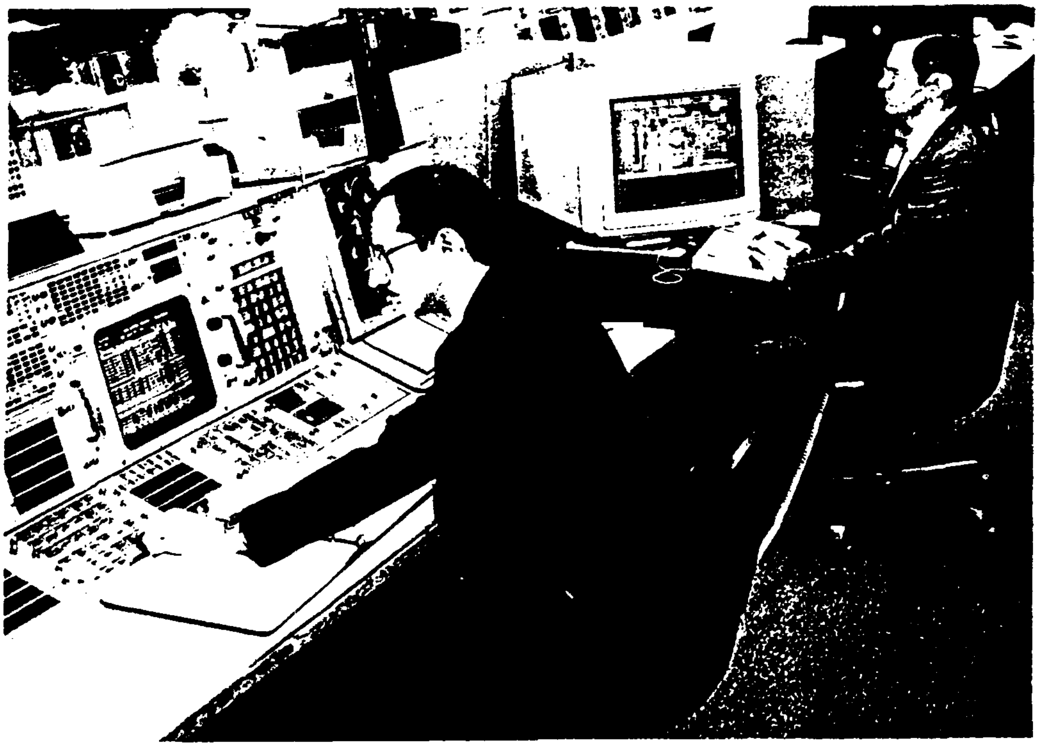
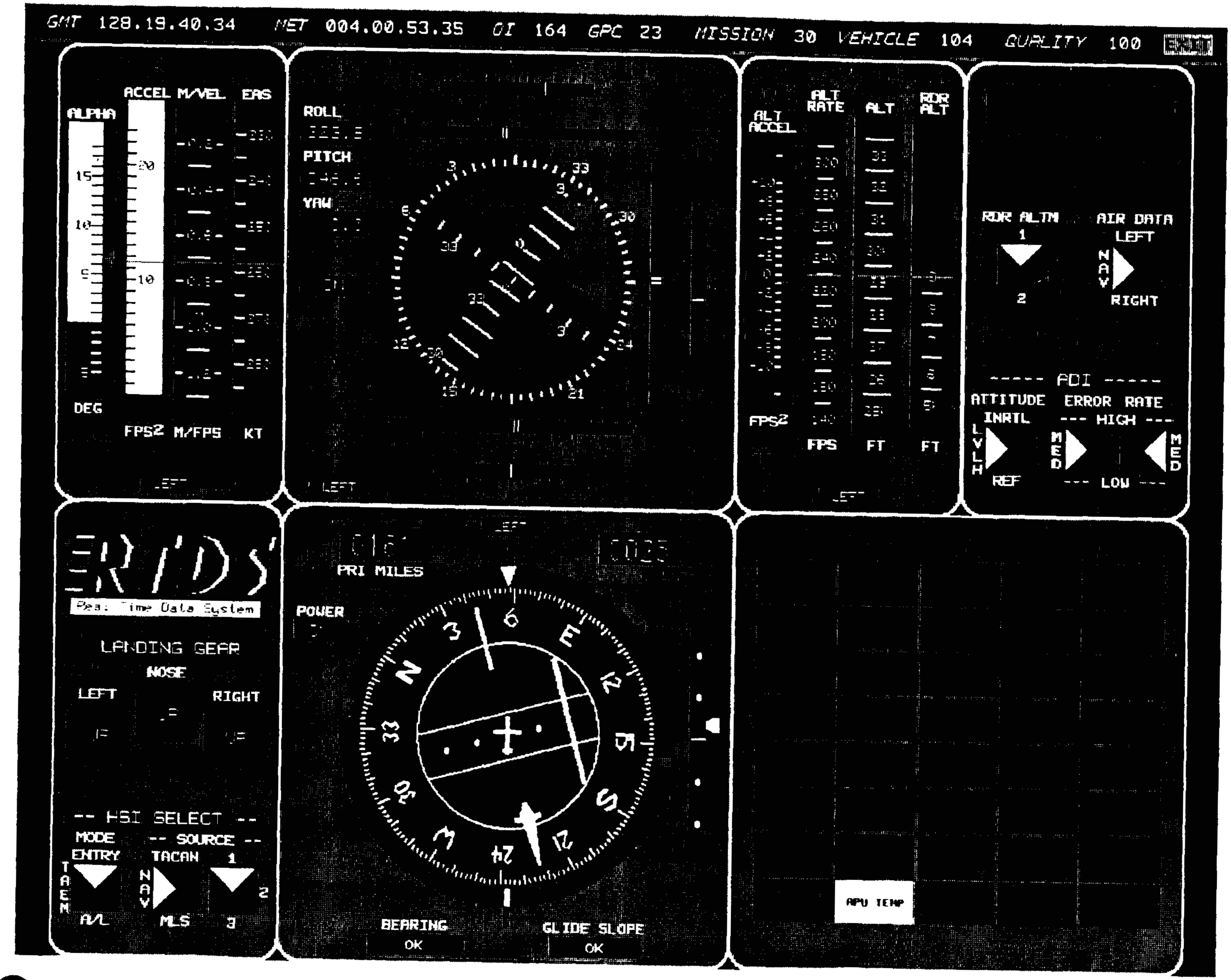
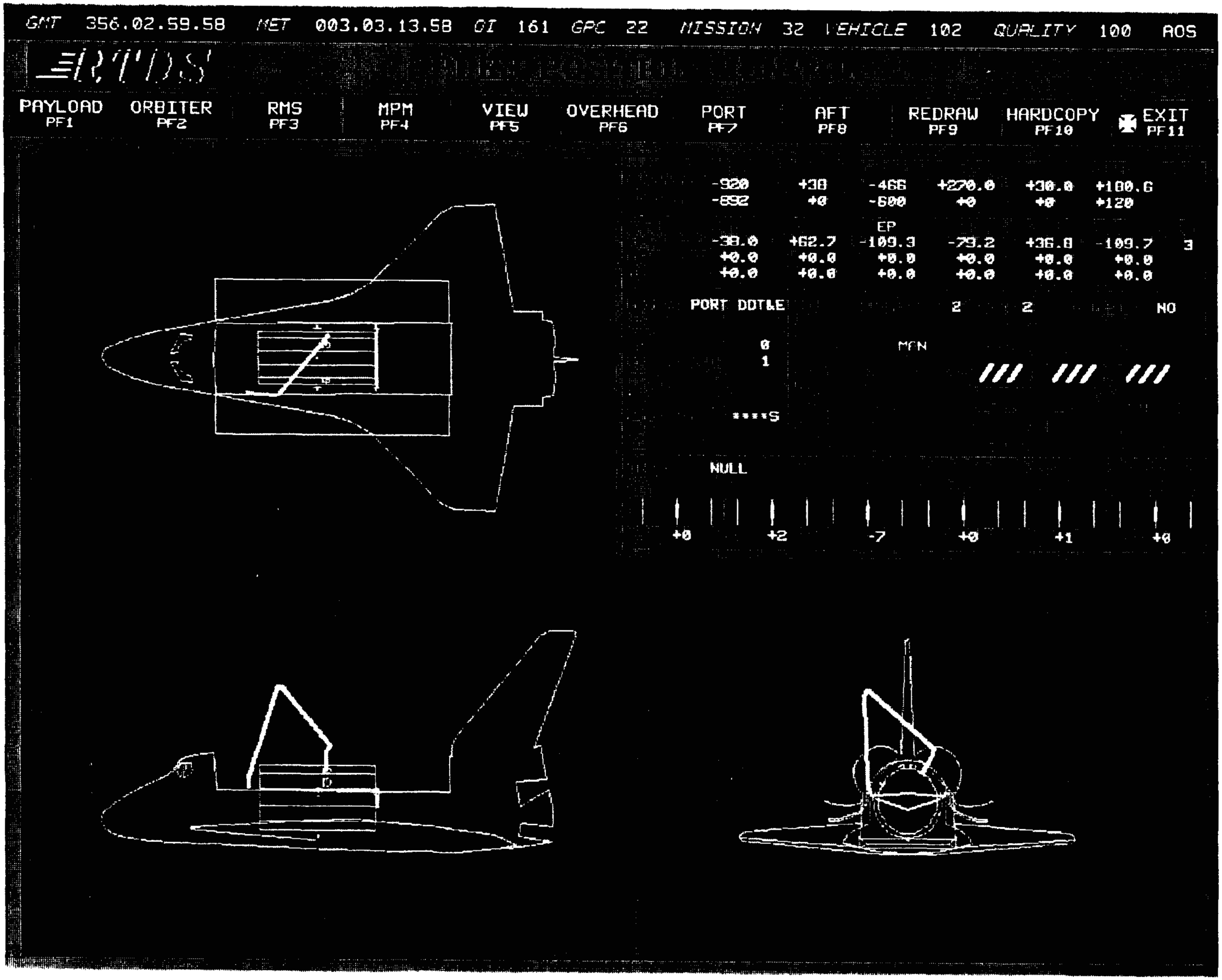


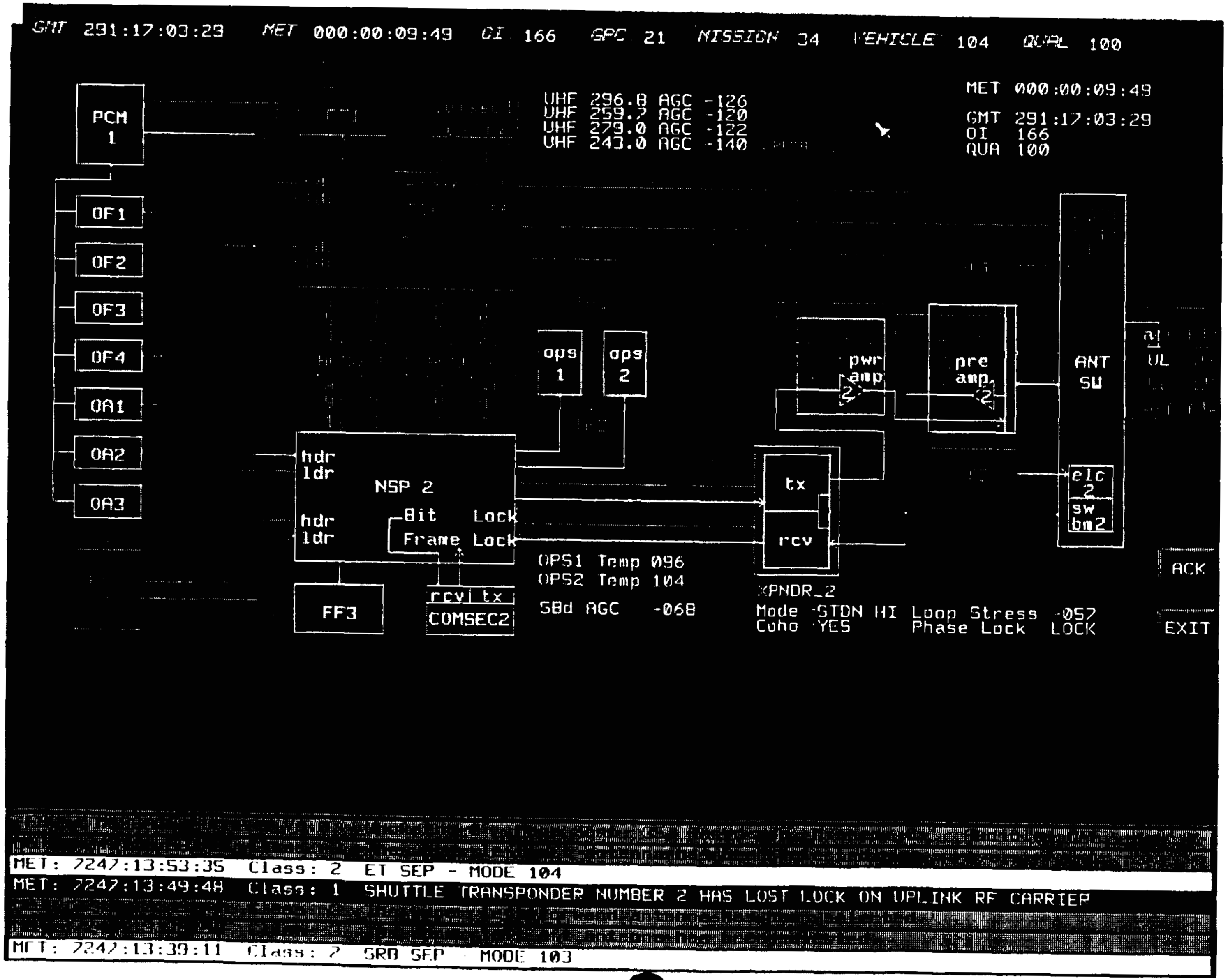
Figure 8.- Conventional INCO workstation (left) and Shuttle INCO Expert System (right).



GMT 128.19.40.34 MET 004.00.53.35 OI 164 GPC 23 MISSION 30 VEHICLE 104 QUALITY 100



```
GMT  356.02.59.58  MET  003.03.13.58  OI  161  GPC  22  MISSION  32  VEHICLE  102  QUALITY  100  AOS

 -920    +38    -466   +270.0   +30.0   +180.6
 -892    +0     -600   +0       +0      +120
                 EP
 -38.0   +62.7  -109.3  -79.2   +36.0   -109.7   3
 +0.0    +0.0   +0.0    +0.0    +0.0    +0.0
 +0.0    +0.0   +0.0    +0.0    +0.0    +0.0

PORT DDTLE                 2       2             NO

       0              MAN
       1                    ///    ///    ///

    ****S

  NULL

   +0      +2      -7      +0      +1      +0
```



GMT 291:17:03:29 MET 000:00:09:49 OI 166 SPC 21 MISSION 34 VEHICLE 104 QUAL 100

UHF 296.8 AGC -126
UHF 259.7 AGC -120
UHF 279.0 AGC -122
UHF 243.0 AGC -140

MET 000:00:09:49

GMT 291:17:03:29
OI 166
QUA 100

OPS1 Temp 096
OPS2 Temp 104

SBd AGC -068

XPNDR_2
Mode STDN HI
Coho YES

Loop Stress -057
Phase Lock LOCK

MET: 7247:13:53:35 Class: 2 ET SEP - MODE 104
MET: 7247:13:49:48 Class: 1 SHUTTLE TRANSPONDER NUMBER 2 HAS LOST LOCK ON UPLINK RF CARRIER
MET: 7247:13:39:11 Class: 2 SRB SEP MODE 103

ARCHITECTURAL KEYS TO INTRODUCING NEW TECHNOLOGY

USING SHADOW STRATEGY

- ISOLATION OF NEW TECHNOLOGY FROM MISSION CRITICAL PROCESSES
- CONNECTIVITY TO REAL TIME DATA
- PLACEMENT IN OPERATIONS LOCATION
- OPERABILITY
- REAL TIME PERFORMANCE
- RAPID CHANGE ACCOMMODATED ("FIREWALL" TO CONTAIN CHANGE)

HARD VS. SOFT REAL TIME

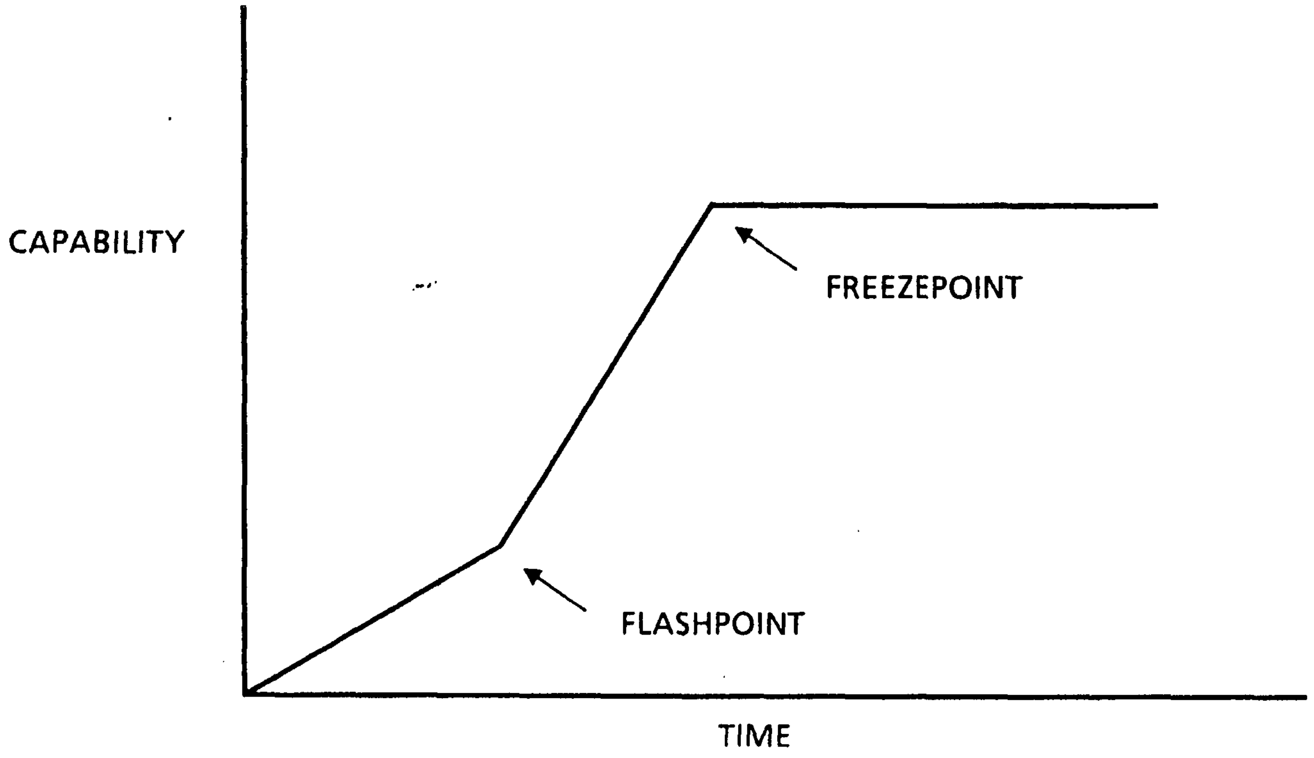
- HARD REAL TIME - IF TIME CONSTRAINTS NOT MET THEN FUNCTION FAILS.
- SOFT REAL TIME - IF TIME CONSTRAINTS NOT MET THEN FUNCTION DEGRADED.
- MOST REAL TIME INTELLIGENT ASSISTANTS MONITORING TASKS ARE SOFT REAL TIME.
- DIFFERENCE BETWEEN HARD AND SOFT REAL TIME NOT WELL-UNDERSTOOD - CAUSES EXCESSIVE DEMANDS/EXPECTATIONS ON SYSTEM DESIGN.
- DISPLAY REQUIREMENT IS TRADITIONALLY ONCE PER SECOND - BUT OPERATIONS DO NOT PERFORM EVALUATIONS AT ONCE/SECOND EVERY SECOND (HARD CONSTRAINT).
- REAL TIME EXPERT SYSTEMS DO NOT HAVE TO RUN 1/SECOND AT ALL TIMES TO HAVE BENEFIT IN MOST (SOFT R/T) APPLICATIONS

  EXAMPLE: 90% 1/SEC
  5% 1/2 SEC
  5% 1/2 SEC
- IS ACCEPTABLE IN TELEMETRY MONITORING TASKS DURING NONDYNAMIC FLIGHT

RTDS - REAL TIME DATA SYSTEM

- INTRODUCED SEVERAL NEW TECHNOLOGIES TO MCC.
- USED OPERATIONALLY ON EIGHT FLIGHTS SO FAR (STS-26 TO STS-32).
  - COLOR GRAPHIC R/T SCHEMATICS AND TELEMETRY BASED ANIMATED VISUALS.
    - RMS (STS-32)
    - FLIGHT INSTRUMENT EMULATION (STS-29)
  - RULE-BASED EXPERT SYSTEMS (INCO, STS-26, GNC STS-32).
  - TASK AUTOMATION (TIRE PRESSURE MONITOR (STS-30).
  - COMMERCIAL OFF THE SHELF TELEMETRY PROCESSOR (STS-26).
  - ETHERNET FOR DATA DISTRIBUTION TO OFFICE ENVIRONMENT (STS-29).
  - THERMAL PRINTING STRIP CHART RECORDERS (STS-29).
  - COLOR HARDCOPY (THERMAL WAX) (STS-30).
  - COMP AND DISPLAY BUILDER TOOLS (STS-26, STS-29).

TECHNOLOGY INSERTION CURVE



FLASHPOINT - WHEN NEW TECHNOLOGY HAS ENOUGH DEMONSTRATED CAPABILITY TO EXCITE USERS

FREEZEPOINT - WHEN USERS RELYING ON NEW TECHNOLOGY AND REQUIRE OPERATIONAL RELIABILITY

CRITICAL REQUIREMENTS

- ABILITY TO RECORD AND PLAYBACK.
- ELECTRICAL ACCESS TO DESIGN INFORMATION.

REAL TIME DATA SYSTEM

BACKGROUND

- MISSION OPERATIONS DIRECTORATE PROJECT THAT UTILIZES:
  - ARTIFICIAL INTELLIGENCE (AI) TECHNIQUES
  - COMMERCIAL OFF-THE-SHELF COMPUTER AND TELEMETRY EQUIPMENT

TO:

- CAPTURE CORPORATE KNOWLEDGE ABOUT SHUTTLE MONITORING
- PRESENT DATA WITH COLOR GRAPHICS TO REDUCE TRAINING, FLIGHT CONTROLLER WORKLOAD AND PROBABILITY OF ERROR
- UPGRADES MISSION CONTROL CAPABILITIES RAPIDLY WITHOUT RISK TO EXISTING COMPLEX
- SUPPORTS LARGER MISSION CONTROL CENTER UPGRADE (MCCU) PLANS BY PROVIDING IMMEDIATE EXPERIENCE WITH CRITICAL TECHNOLOGIES

Real Time Data Systems: Incorporating New Technology in Mission Critical Environments

John F. Muratore
Troy A. Heindel

National Aeronautics and Space Adminstration
Mission Operations Directorate
Lyndon B. Johnson Space Center

BACKGROUND

Real Time Data System (RTDS) is a Mission Operations Directorate (MOD) project that utilizes Artificial Intelligence (AI) and commercial off-the-shelf computer and telemetry equipment to: (1) capture corporate knowledge about shuttle monitoring, (2) present data with color graphics to reduce operator training time, and (3) reduce flight controller workload and probability of error. RTDS has upgraded Mission Control capabilities rapidly without risk to the existing complex. Additionally, RTDS supports the larger Mission Control Center Upgrade (MCCU) plans by providing immediate experience with critical technologies such as real time expert systems.

RTDS has been used by flight controllers in Mission Control since the flight of Discovery in September 1988. Since that time it has expanded in size and scope providing an operational testbed for promising new technolgies and transitioning proven technologies to flight operational status for addition to the flight controller tool base.

REAL TIME DATA SYSTEM

APPROACH

- ISOLATE NEW TECHNOLOGY FROM CRITICAL SYSTEMS
- CONNECT TO REAL-TIME DATA
- GENERATE RESULTS IN REAL TIME
- LOCATE EQUIPMENT AT FLIGHT CONTROLLER POSITIONS
- "FIREWALL" SOFTWARE TO ISOLATE THE EFFECTS OF CHANGE AND ALLOW RAPID PROTOTYPING

APPROACH

In past programs, NASA managers have been reluctant to rely on new technologies for mission critical activities until they are proven in non-critical areas. With RTDS, NASA-MOD has developed a non-traditional method for migrating the new technologies more rapidly into the operator tool base by "field testing" them in the mission critical environment of Mission Control. This approach mandates several important requirements.

- **ISOLATE NEW TECHNOLOGY FROM CRITICAL SYSTEMS**

Isolatation is a key ingrediant when hosting new technologies within an operational environment. The new technology must be isolated from existing systems to avoid inducing problems with the trusted system. On the flip side, the problems or deficiencies of the previous system should not be induced into the new technology through design dependancies. The strategy of isolation works especially well in Mission Control as the older system is main frame based and not espesially forgiving of change and does not easily connect to the new workstation platforms.

- **CONNECT TO REAL-TIME DATA**

Connectivity to real time data is an absolute necessity; real time data is to the flight controller as lumber is to the carpenter. The characterstics of the data will drive the design of the overall system as well as the individual applications.

APPROACH (CONT.)

- **GENERATE RESULTS IN REAL TIME**

In Mission Control the operator must see data and results of computer based monitoring applications in real time. The definition of real time changes from system to system, but in generic terms means the ability to generate results in time for the operator to effect corrective changes in system configuration based on those results.

- **LOCATE EQUIPMENT AT FLIGHT CONTROLLER POSITIONS**

Laboratory prototyping is a good way to experiment with new technologies, but is a "gee-wiz" activity as far as line flight controllers are concerned. If the flight controller can't test-drive the new technology in his natural environment, the really important lessons are not discovered until its too late to effect the necessary changes. Issues of operability arise under the stress of real operations.

- **"FIREWALL" SOFTWARE TO ISOLATE THE EFFECTS OF CHANGE AND ALLOW RAPID PROTOTYPING**

Complex data systems have to be tolerant of change. Change is motivated by advances in the technology, a desire to add new capabilities, or the need to modify existing ones. The software life cycle chosen by the RTDS team placed further requirements on the ability of the system to withstand change. Selected applications are used operationally in a mission critical environment; these require a high degree of host system stability. Other less mature applications may require additional system level capabilities not found in the previous host systems. By "firewalling" the system, and maintaining upward compatibility the overall system is made tolerant of change.

REAL TIME DATA SYSTEM

RESULTS

- SIGNIFICANT KNOWLEDGE CAPTURED IN AUTOMATED MONITORING
- ABLE TO DO MORE WORK WITH SAME RESOURCES
  - INCO TEAM BEING REDUCED FROM FOUR TO THREE IN LATE FY90
- RTDS PROCESSES DATA 3-4 SECONDS FASTER THAN MAIN FRAME
- UPGRADING CAPABILITIES IN PARALLEL WITH ONGOING OPERATIONS WITH NO RISK TO FACILITY
  - REPLACED THREE DISPLAY CONSOLES WITH RTDS DISPLAYS
  - INSTALLED EIGHT RTDS DISPLAYS ON CONSOLES
  - ATTACKING BACKLOG OF POSTPONED RQMTS IN MAINFRAME BY PLACING THEM IN WORKSTATIONS
  - IMPLEMENTED 6 ALGORITHMS IN WORKSTATIONS THAT HAD MAINFRAME ESTIMATED COSTS OF $10-100K EACH
  - SOFTWARE TOOLS DEVELOPED IN RTDS HAVE BEEN BASELINED IN MISSION CONTROL CENTER UPGRADE

RESULTS

- **SIGNIFICANT KNOWLEDGE CAPTURED IN AUTOMATED MONITORING**

RTDS employs both algorithmic and heuristic knowledge representation schemes. C language code and rules are used for those tasks requiring a high rate of execution. Rule based systems such as the C Language Inference Production System (CLIPS by NASA-JSC) and Gensym's G2 software are used to capture more complex determistic and heuristic knowledge. Over three hundred real time algorithms have been developed and tested and are used routinely by flight controllers in Mission Control for fault detection. Real time expert systems have been developed and operated in several disciplines including Integrated Communications Officer (INCO), and Guidance, Navigation and Control (GNC). The majority of these applications could not have been developed on the existing main frame system; the ones that could would have been done at considerably greater cost.

- **ABLE TO DO MORE WORK WITH SAME RESOURCES**

Prior to RTDS the flight controller had to keep eyes on the data, since little or no automated monitoring is done by the main frame. Through automated monitoring and fault detection RTDS has provided a means for the flight controller to work other issues such as mission planning. If a problem does arrise, the flight controller is notified by RTDS.

- **INCO TEAM BEING REDUCED FROM FOUR TO THREE**

The INCO flight controllers are using RTDS capabilities to automate one of their back room positions. The Data Communications (DATACOMM) Officer is responsible for monitoring and controlling the Orbiter's flight recorders. A preliminary system will be tested in early Summer, with the full-up system ready by the end of the year. Though the DATACOMM expert system will not initially have command capability to the orbiter it will automate the bulk of the DATACOMM's activities which include data and system configuration management, fault detection and resolution.

RESULTS (CONT.)

- **RTDS PROCESSES DATA 3-4 SECONDS FASTER THAN MAIN FRAME**

Using commercial off-the-shelf telemetry processing equipment and general purpose engineering workstations RTDS is able to put processed real time data up on a display 3-4 seconds faster than the main frame. During the dynamic phases of flight those few seconds can be used to make higher quality decisions which can affect the successful outcome of the flight.

- **UPGRADING CAPABILITIES IN PARALLEL WITH ONGOING OPERATIONS WITH NO RISK TO FACILITY**

One of the prime concerns with installing new technology into a mission critical area is how it affects existing capabilities. RTDS took the standpoint early on that the new technology had to be installed in the Mission Control Center, but isolated from the existing main frame complex. In this way, RTDS does not interact in any way with the existing system. The existing system remains in place until sufficient confidence has been built in the new system.

- **REPLACED THREE DISPLAY CONSOLES WITH RTDS DISPLAYS**

Three RTDS displays have been placed inside of flight controller consoles displacing the main frame displays. This represents the first time in almost twenty years that the existing console hardware has been removed to accomadate newer technology. Replacement was done after extensive side-by-side testing in simulations and flight. This replacement represents a significant step in the upgrade process.

- **INSTALLED EIGHT RTDS DISPLAYS ON CONSOLES**

Eight RTDS displays have been placed on or next to flight controller consoles. These displays are used for side-by-side testing of operational and near operational real time applications during simulations and flight. Flight controllers use these displays routinely to build confidence in their new operator tools.

RESULTS (CONT.)

- **ATTACKING BACKLOG OF POSTPONED RQMTS IN MAINFRAME BY PLACING THEM IN WORKSTATIONS**

There currently exists a backlog of requirements waiting to be implemented in Mission Control's main frame complex. Each new main frame requirement which is instituted costs many thousands of dollars and takes a team of programmers to implement. There is a natural reluctance to implement these requirements for fear of adversely affecting existing capabilities. RTDS has implemented many of these requirements. The Main Engine application is a good example of how RTDS has implimented new capabilities in rapid fashion. The Main Engine application was developed and certfied for flight in three months.

- **SOFTWARE TOOLS DEVELOPED IN RTDS HAVE BEEN BASELINED IN MISSION CONTROL CENTER UPGRADE**

RTDS has developed many tools and techniques which did not previously exist on any other systems or in the commercial world. The Computation Development Environment (CODE) was developed in RTDS to enable flight controllers with little or no programming experience to implement real time C language algorithms. CODE has since been baselined for use in the larger Mission Control Center Upgrade project, the Space Station Control Center, and the Multi-Purpose Control Center.

Providing time homogenous real time telemetry data to real time applications on multi-tasking workstations poses some interesting challenges. RTDS has developed an innovative multi-buffer design to accomadate the requirements of the data and the characteristics of the UNIX operating system. This design and the driver routine have been adopted by Mission Control Center Upgrade project.

REAL TIME DATA SYSTEM

PRODUCTS

INCORPORATED INTO OPS TOOL BASE

- MAIN ENGINE WORKSTATION MORE CAPABLE THAN MAINFRAME (SINCE STS-26)
- INCO SYSTEM HAS SIGNIFICANTLY REDUCED TIME TO FAULT DETECTION (SINCE STS-26)
  - DETECTED FAILURES DURING STS-34, STS-32
- RMS APPLICATION USES REAL TIME ANIMATION TO PROVIDE REAL-TIME VISUALIZATION AND ERROR DETECTION (STS-32)
  - USED EXTENSIVELY DURING LDEF GRAPPLE & PHOTO SURVEY
- FLIGHT INSTRUMENTS USED SINCE STS-29
- MECHANICAL SYSTEMS TIRE PRESSURE APPLICATION

  (STS-34, STS-33, STS-32)
- RTDS TELEMETRY CAPABILITY IS PORTABLE AND PROVIDES EMERGENCY MISSION CONTROL CENTER CAPABILITY (TESTED DECEMBER 1987 & DURING STS-32)
- RTDS TELEMETRY AND REAL TIME EXPERT SYSTEM SOFTWARE TRANSFERRED TO OAST AERONAUTICS (INTEGRATED TEST FACILITY AT DFRF) AND USAF (F-15 STOL PROJECT AT EAFB)
- REAL TIME SOFTWARE USED BY JSC ENGINEERING DIRECTORATE FOR PRE-LAUNCH INERTIAL MEASUREMENT UNIT EVALUATION

PRODUCTS

INCORPORATED INTO OPS TOOL BASE

- **MAIN ENGINE WORKSTATION MORE CAPABLE THAN MAINFRAME (SINCE STS-26)**

In May of 1988 the flight controllers responsible for monitoring the main engines determined that there were several failure modes of the main engine which required automated monitoring. The flight controllers could not manually perform the calculations and assessments fast enough to meet the demands of monitoring this high performance system in dynamic flight. The necessary fault-detection routines were designed and built using RTDS. The system was certified for use in August 1988 and was used operationally during STS-26 in September 1988. The system has since been expanded and re-certified several times in order to handle other automated monitoring tasks which are not done in the main frame.

- **INCO SYSTEM HAS SIGNIFICANTLY REDUCED TIME TO FAULT DETECTION (SINCE STS-26)**

During simulations in mission control the INCO system routinely detects failure conditions before the operator can detect them on the main frame display. The main frame display system can not display all the data about all the systems simulataneously. Therefore the data which indicate a failure might not be visible to the flight controller at the time of the failure. The algorithms built into the INCO system check hundreds of parameters each second. If these algorithms detect a failure condition they annunciate on the RTDS display with a color coded message. The INCO algorithms were designed and implimeted by INCO flight controllers.

PRODUCTS (CONT.)

- **DETECTED FAILURES DURING STS-34, STS-32**

During STS-34 the INCO system detected a fault with the Orbiter's Sband quad antennas and annunciated it on the RTDS display. This failure condition, though visible on the main frame display, was detected first by flight controllers monitoring the RTDS display.

During STS-32, multiple failures of the Orbiter's Text and Graphics System (TAGS) where detected first by flight controllers monitoring the RTDS display.

- **RMS APPLICATION USES REAL-TIME ANIMATION TO PROVIDE REAL-TIME VISUALIZATION AND ERROR DETECTION (STS-32)**

The Remote Manipulator System (RMS) application is aptly called the Position Monitor. The Position Monitor uses real time data to dynamically display the position of the Orbiter's Remote Manipulator System or Arm. Position Monitor allows the flight controllers to visualize the position and movements of the Arm in real time. As the Arm does not have a collision avoidance system real time visualization is especially important.

- **FLIGHT INSTRUMENTS USED SINCE STS-29**

- **MECHANICAL SYSTEMS TIRE PRESSURE APPLICATION (STS-34, STS-33, STS-32)**

The Tire Pressure application provides the Mechanical Systems flight controllers with real time trend analysis of the Orbiter's tires. Similar to the RMS application, Orbiter tire pressures were previously typed into a portable computer and graphed using Lotus 1-2-3. The data is now automatically logged and plotted on their RTDS display, freeing up the flight controllers to do more important tasks.

PRODUCTS (CONT.)

- **RTDS TELEMETRY CAPABILITY IS PORTABLE AND PROVIDES EMERGENCY MISSION CONTROL CENTER CAPABILITY (TESTED DECEMBER 1987 & DURING STS-32)**

The RTDS telemetry system is portable and rugged. The RTDS telemetry system has been used to provide real time data for the Emergency Mission Control Center since 1987. In the event of a natural disaster occuring at JSC in which all monitoring capabilities were lost at the Mission Control Center a small team of flight controllers would be flown out to White Sands, New Mexico equipped with the RTDS system. The data and displays provided by RTDS would be used by flight controllers to calculate proper trajectory for de-orbit and safe return.

- **RTDS TELEMETRY AND REAL TIME EXPERT SYSTEM SOFTWARE TRANSFERRED TO OAST AERONAUTICS (INTEGRATED TEST FACILITY AT DFRF) AND USAF (F-15 STOL PROJECT AT EAFB)**

- **RTDS REAL TIME SOFTWARE USED BY JSC ENGINEERING DIRECTORATE FOR PRE-LAUNCH INERTIAL MEASUREMENT UNIT EVALUATION**

REAL TIME DATA SYSTEM

PRODUCTS

TESTING PHASE

- REAL TIME EXPERT SYSTEMS DEVELOPED USING COTS REAL TIME EXPERT SYSTEM TOOL (GENSYM'S G2)
    - GNC AIR DATA PROBES DURING STS-34, STS-32
    - INCO FLIGHT RECORDERS DURING STS-34, STS-32
    - GNC CONTROLLABILITY DURING STS-32
- LANDING SITE SELECTION APPLICATION TESTED IN WEATHER OFFICE DURING STS-32
- PAYLOAD BAY DOOR APPLICATION MONITORS DATA NOT YET AVAILABLE ON MAINFRAME (STS-32)
- DEMONSTRATED REMOTE MONITORING CAPABILITY

PRODUCTS

TESTING PHASE

- **REAL TIME EXPERT SYSTEMS DEVELOPED USING COTS REAL TIME EXPERT SYSTEM TOOL (GENSYM'S G2)**

G2 is a commercial off-the-shelf real time rule based expert system shell which has previously been used in process control applications. The software is a product of the Gensym Corporation. The software provides many of the robust knowledge acquistion capabilities necessary for building and maintaining distributed real time applications. G2 has been used by RTDS to rapidly develop several real time expert systems.

- **GNC AIR DATA PROBES DURING STS-34, STS-32**

The Guidance, Navigation & Control (GNC) Officer's Air Data Probe expert system was developed to monitor the Orbiter's two air data probes during descent. The system has been tested during STS-34 and STS-32. Air Data Probe expert system represents RTDS's first use of commercial off-the-shelf expert system tools.

- **INCO FLIGHT RECORDERS DURING STS-34, STS-32**

The INCO Ops Recorders expert system will be used as the basis for position automation at the DATACOMM position. The DATACOMM is the flight controller responsible for operating the Orbiter's flight recorders.

- **GNC CONTROLLABILITY DURING STS-32**

The GNC Controllability expert system has been developed to monitor the Orbiter's computer control during powered flight. This system was tested during STS-32.

PRODUCTS (CONT.)

- **LANDING SITE SELECTION APPLICATION TESTED IN WEATHER OFFICE DURING STS-32**

Weather data concerning Shuttle runways is currently recieved by the Mission Control Center weather office and verbally conveyed to the flight director over the MCC voice loops. These voice loops are very busy during the dynamic phases of flight and information transfer via voice loop can become difficult. The Landing Site Selection application recieves the real time data electronically and displays it on a color graphics workstation. One of these workstations has been installed in the JSC Weather office and was tested during STS-32. A second workstation will be installed this summer at the Flight Director console. This will provide the Flight Director direct access to the weather information and cut down on voice loop traffic during dynamic phases of flight.

- **PAYLOAD BAY DOOR APPLICATION MONITORS DATA NOT YET AVAILABLE ON MAINFRAME (STS-32)**

The Payload Bay Door application is another RTDS only capability. The data provided by this display is not available on the main frame and will not be for some years to come. This application was tested during STS-32 and will be made operational by STS-35.

- **DEMONSTRATED REMOTE MONITORING CAPABILITY**

RTDS has developed the capability to transmit real time data via standard Ethernet and data phones. These capabilities were originally designed to provide an office monitoring capability for flight controllers. The capabilities have since proven invaluable in providing data to applications developers who do not have direct access to RTDS telemetry processing equipment.

REAL TIME DATA SYSTEM

FY90 ACTIVITIES

- EXTEND RTDS SUPPORT TO DPS, FLIGHT DIRECTOR, EGIL, EECOM AND POINTING FLIGHT CONTROL DISCIPLINES (STS-35)
- VISION SYSTEM TO ASSIST ORBITAL MANEUVERING VEHICLE (OMV) PILOT GUIDE OMV DURING PROXIMITY OPERATIONS (FALL 1990)
- DISTRIBUTED COOPERATIVE EXPERT SYSTEM FOR BUS LOSS (STS-37)
- EXPAND REMOTE MONITORING CAPABILITY BY PROVIDING ACCESS RTDS DISPLAYS BY FLIGHT CONTROLLER OFFICE COMPUTERS (STS-37)

FY90 ACTIVITIES

- **EXTEND RTDS SUPPORT TO DPS, FLIGHT DIRECTOR, EGIL, AND EECOM CONTROL DISCIPLINES (STS-35)**

- **VISION SYSTEM TO ASSIST ORBITAL MANEUVERING VEHICLE (OMV) PILOT GUIDE OMV DURING PROXIMITY OPERATIONS (FALL 1990)**

The Orbital Maneuvering Vehicle (OMV) shall be limited to two monochrome TV cameras during proximity operations. In order to assist the OMV Pilot in docking procedures a vision system will be developed which uses video input to determine range and attitude. The system shall be installed in the OMV Control Center. RTDS is working closely with vision system experts at the Ames Research Center to develop this capability.

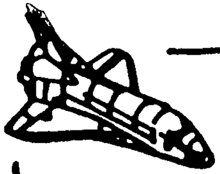
- **DISTRIBUTED COOPERATIVE EXPERT SYSTEM FOR BUS LOSS (STS-37)**

RTDS will develop its first distributed cooperative expert system to monitor Orbiter bus loss. Orbiter Bus Loss has been chosen because the effects of bus loss are felt across multiple flight control disciplines. The problem is well understood and well documented. Orbiter bus loss is currently the responsibility of the EGIL flight controllers.

TRANSITION FLIGHT CONTROL ROOM AUTOMATION

by

Curtis Ray Welborn
National Aeronautics and Space Administration,
Lyndon B. Johnson Space Center
Mail Code FS72
NASA Road 1
Houston, Texas 77258



TRANSITION FLIGHT CONTROL ROOM AUTOMATION

Curtis Ray Welborn
NASA/Johnson Space Center

February 1990

ABSTRACT

The Workstation Prototype Laboratory is currently working on a number of projects which we feel can have a direct impact on ground operations automation. These projects include:

- The Fuel Cell Monitoring System (FCMS), which will monitor and detect problems with the fuel cells on the Shuttle. FCMS will use a combination of rules (forward/backward) and multi-threaded procedures which run concurrently with the rules, to implement the malfunction algorithms of the EGIL flight controllers. The combination of rule based reasoning and procedural reasoning allows us to more easily map the malfunction algorithms into a real-time system implementation.

- A graphical computation language (AGCOMPL). AGCOMPL is an experimental prototype to determine the benefits and drawbacks of using a graphical language to design computations (algorithms) to work on Shuttle or Space Station telemetry and trajectory data.

- The design of a system which will allow a model of an electrical system, including telemetry sensors, to be configured on the screen graphically using previously defined electrical icons. This electrical model would then be used to generate rules and procedures for detecting malfunctions in the electrical components of the model.

- A generic message management (GMM) system. GMM is being designed as a message management system for real-time applications which send advisory messages to a user. The primary purpose of GMM is to reduce the risk of overloading a user with information when multiple failures occurs and in assisting the developer in devising an explanation facility.

The emphasis of our work is to develop practical tools and techniques, while determining the feasibility of a given approach, including identification of appropriate software tools to support research, application and tool building activities.

AGENDA

INTRODUCTION

FISCAL YEAR 1989

FISCAL YEAR 1990

ARMOA PROJECT

- Mission Operations Support Study (MOSS)
- Alternate Language Interface (ALI)
- Generic Message Management (GMM)

INTRODUCTION

This is a Code S RTOP sponsored by Gregg Swietek from NASA Headquarters. The work for the Transition Flight Control Room has been conducted by the Workstation & Visual Systems Branch which is part of the Systems Development Division. In fiscal year 1989 the team members were Allen Brewer, (NASA/Section Head of the Workstation Systems Development Section), Clark Pounds, (NASA/Lab. manager for the Workstation Prototype Laboratory), Danny Labasse (MITRE) and Dave Hammen (MITRE). For fiscal year 1990 team members are Allen Brewer, Curtis Welborn (NASA), Frederic Gibbs (NASA), Charlie Robertson (McDonnell Douglas), Wayne Parrott (LinCom) and Yashvant Jani (LinCom). The objectives of the Transition Flight Control Room are characterized in the following paragraphs.

> At some point in the prototyping process, it is necessary to test the software using operational data. Such testing is difficult in an operational environment characterized by strict controls that permit only qualified software to execute. In a near-operational environment, in which some of the strict controls are removed, near-operational data can be fed to prototype software and aid the prototyping process. The Transition Flight Control Room (TFCR) provides such an environment, allowing control center prototypes to be tested using operational data. NASA's Hardware Independent Software Environment (HISE) provides standardized tools and rules for software developed for the TFCR and related workstation laboratories, but does not yet provide support for advanced automation techniques.
>
> The goal of the TFCR advanced automation task is to augment the HISE with appropriate tools and techniques so that advanced automation software may be developed within the HISE and used in the TFCR.[1]

INTRODUCTION

- Test Flight Control Room Automation Task (TFCR)
  Code S RTOP : Gregg Swietek

- JSC MSD/SDD/Workstation & Visual Systems Branch/
  Workstation Systems Development Section (FS72)

- FY 1989 Members
  (NASA) Allen Brewer (Section Head), Clark Pounds
  (MITRE) Daniel Labasse, David Hammen
  (Ford Aerospace) John Engvall, Matt Hanson, Charles Copeland

- FY 1990 Members
  (NASA) Allen Brewer, Curtis Ray Welborn, Frederic Gibbs
  (LinCom) Wayne Parrott, Yashvant Jani
  (McDonnell Douglas) Charlie Robertson

TFCR's goal is to augment NASA's Hardware Independent Software Environment with tools and techniques which can be used in automating the control center environments (SSCC/MCC)

Fiscal Year 1989

For most of 1989 the MITRE Corporation conducted interviews and worked to produce a report describing functional requirements, system requirements, and selection factors for the advanced automation tools to be acquired for the TFCR/HISE. The addition of 7 methodologies into the TFCR were recommended in the final MITRE report. The 7 methodologies were : Rule-Based Reasoning, Hypermedia, Object-Oriented Programming, Model-Based Reasoning, Databases, Voice Generation and Computer-Aided Software Engineering Tools. In addition to the 7 methodologies recommended by MITRE two methodologies, Neural Networks and Analogical (Case-Based) Reasoning, were mentioned as future additions to the TFCR.

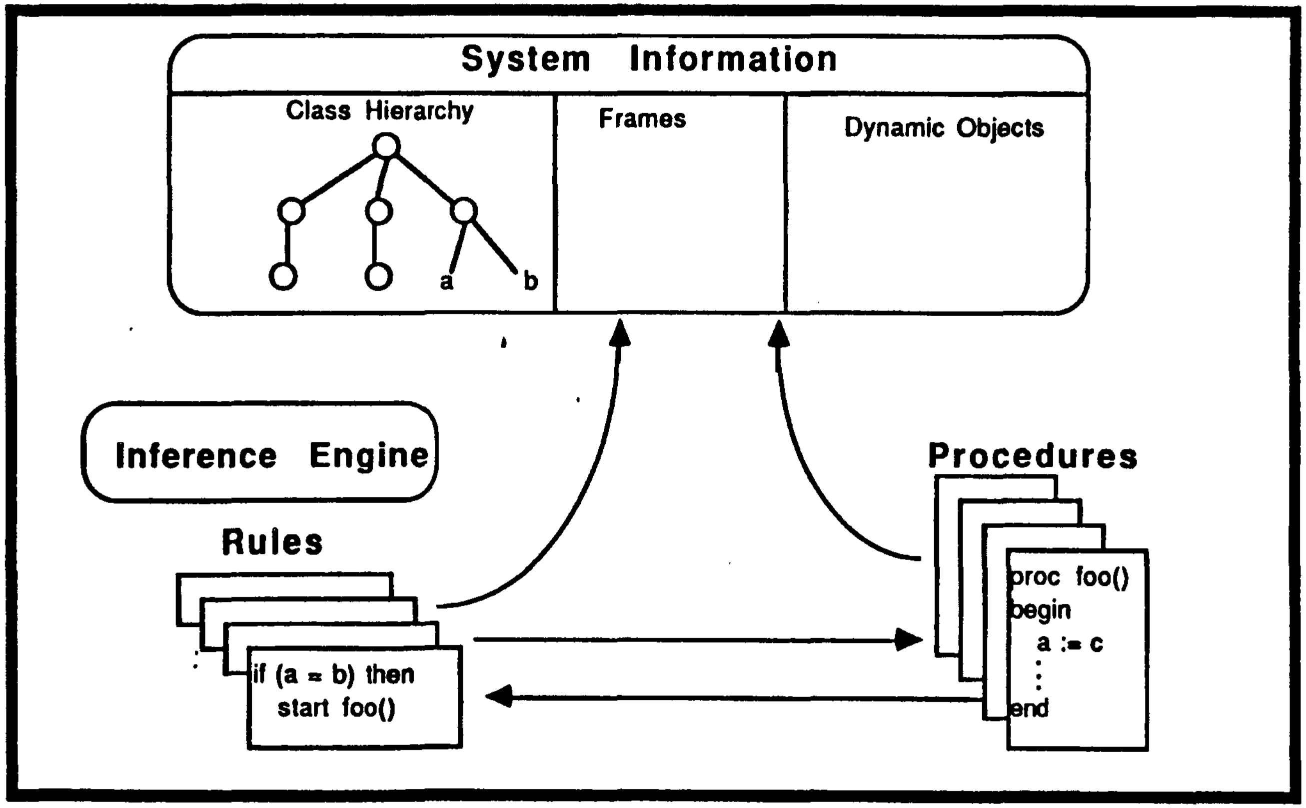
The current members of this RTOP strongly suggest the addition of some form of Procedure-Based Reasoning to augment existing Rule-Based Reasoning systems. Our desire for Procedure-Based Reasoning is driven by the need to execute **multiple** procedural algorithms (e.g. malfunction procedures) concurrently within a Rule-Base environment. Because of the need to share information between procedural algorithms and rules, an environment which intergrates both rules and procedures offers the best development and maintaince environments. This ability would allow concurrent execution of any number of algorithms while continuing to perform data driven monitoring of the systems' health and status.

FISCAL YEAR 1989

- MITRE Final Report
  - Recommended Methodologies to the TFCR
    Rule-Based Reasoning
    Hypermedia
    Object-Oriented Programming
    Model-Based Reasoning
    Databases
    Voice Generation
    Computer-aided Software Engineering Tools
  - Functional Requirements
    Developed for Recommended Methodologies

FISCAL YEAR 1989 (continued)

••Additional Methodologies
  Neural Networks
  Analogical (Case-Based) Reasoning
  Procedure-Based Reasoning



Fiscal Year 1990 Work

As of this fiscal year a new project within the TFCR task has started, Applied Research in Mission Operation Automation (ARMOA) task. The objectives of this project are to evaluate and construct software systems which will aid in the automation or documentation of process within the control centers and training facilities. This is to be accomplished by automating and/or easing the acquisition of knowledge, the design of knowledge structures and the development, validation and improvement of expert systems. Three different project areas currently exist within ARMOA: MOSS, ALI, and GMM. While the projects are distinct in nature, there is a great deal of information shared between the projects. The MOSS project, due to its operational nature, is currently being used to direct most of the research and implementations in our other projects.

MOSS

MOSS, the Mission Operations Support Study, is a development project in which an operational system will be developed and studied. By studying both the technical and human issues within the current control center, we are better able to direct our research in how we can construct new systems for the Space Station. The major activity of MOSS is currently the construction of the Fuel Cell Monitoring System (FCMS), a health and status monitoring system for the EGIL flight controllers. The system is being directed at, but not limited to, monitoring and detecting problems with the Fuel Cells onboard the Orbiter. By actively pursuing the development of an operational system, while studying the operational environment it must work within, we are gaining valuable insights into what is needed and what is wanted. Two major areas of study exist for us while developing FCMS: Environmental Studies and Technology Evaluation.

In our Environmental Studies we are most interested with how the current job of monitoring gets done and what the users (flight controllers) want in new systems. This involves understanding issues related to:

- operating and constructing real-time health and status monitoring systems;
- dealing with long duration monitoring in the face of computer system failures, the reconfigurations of equipment due to physical changes or break downs;
- technology transfer issues such as user interfacing, implementing active vs. passive resource management systems, and the development of user trust for these new technologies.

Technology evaluation is our second primary area of interest. We are currently evaluating the use of G2, a real-time expert system development environment and the relationship between rules and algorithms for implementing an operational system.

FISCAL YEAR 1990

•Applied Research in Mission Operation Automation (ARMOA)

••Mission Operations Support Study (MOSS)

•••Operational Project
EGIL Fuel Cell Monitoring System (FCMS)

•••Environmental Studies
Real-time Health & Status Monitoring
Long Duration Health & Status Monitoring
System Failure Recoveries
Reconfiguration Dynamics
Technology Transfer
Passive/Active System Management
Interface Designs

•••Technology Evaluation
G2
Rules vs. Procedures

ALI

ALI (Alternate Language Interface) is our second major project. The three subprojects currently being worked are centered around the basic theme of capturing knowledge from a user by supplying the user with languages which allow them to encode their knowledge more easily. These subprojects are:

- ATA (Alarms Triggers & Algorithms) is a software architecture and design philosophy which is being defined for real-time monitoring systems. Syntax and semantics to support ATA are also being defined, though we have no plans to implement this architecture. We would prefer, when finished with the architecture, to present it to existing commercial developers for inclusion into their products. ATA has been, and is continuing to be, heavily influenced by the Procedural Reasoning System (PRS) from SRI International and G2 from Gensym Corp. Both G2 and PRS support a form of Procedure-Based Reasoning though their implementations differ greatly.

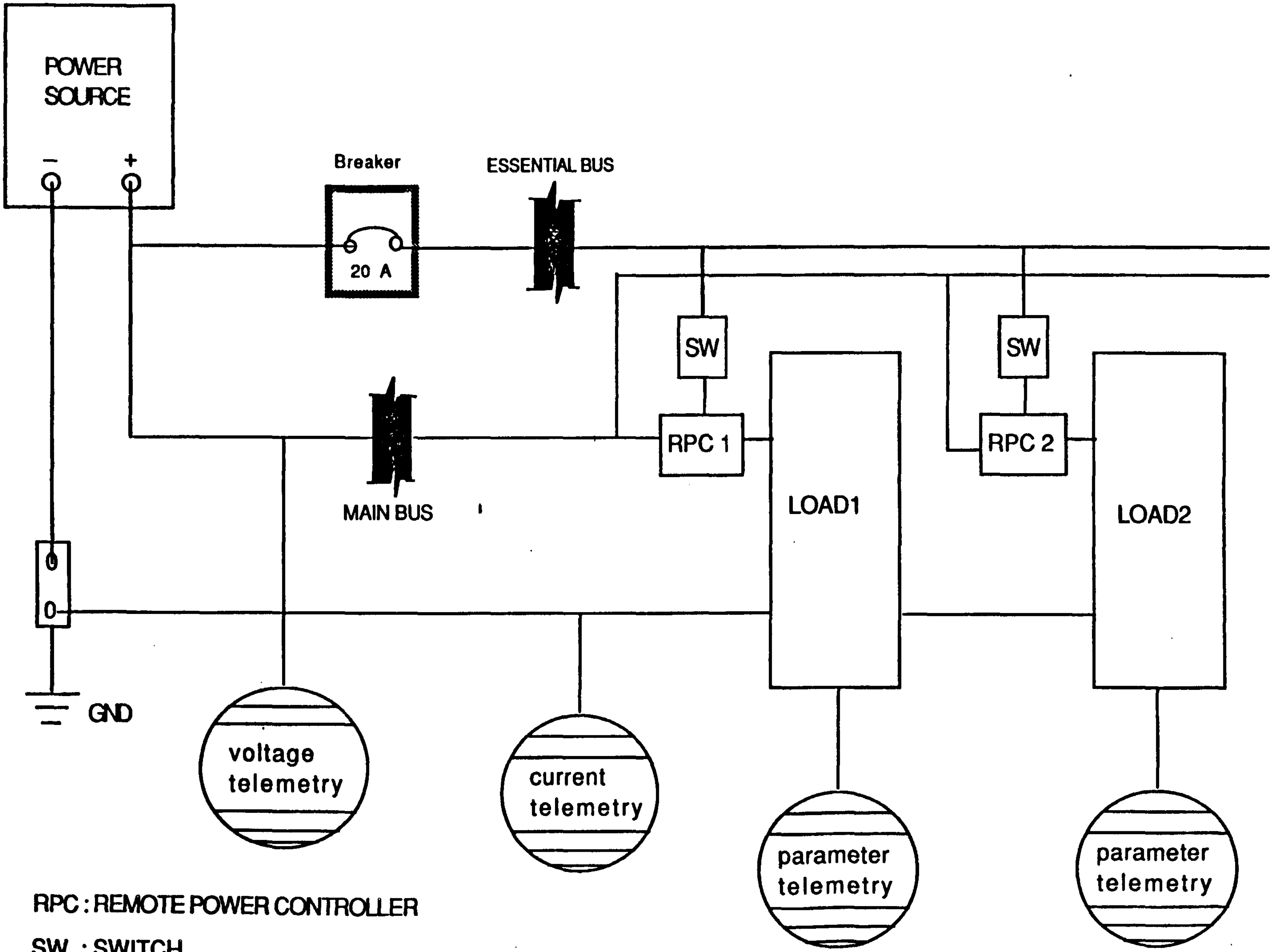
- AGCOMPL (A Graphical Comp Builder) is a graphical programming prototype that produces source code for the MSD COMP Builder using the extended MOLE grammar. The graphical language design of AGCOMPL was derived from circuit diagrams, where every operation to be performed has a unique icon, (e.g. AND GATE and OR GATE in circuit diagrams vs. EQUAL GATE and ADD GATE in AGCOMPL). AGCOMPL has been designed for non-programmers or people with little programming experience. Phase 1 of AGCOMPL will be completed this month, with an evaluation process to follow. Commercial products which support this same form of graphical programming are being reviewed in the Workstation Prototype Laboratory.

- MEPS (Modeling of Electrical Power Systems) is one of a set of projects relating to the modeling of physical devices. The overall objective of these projects is to provide a modeling environment capable of producing rules or algorithms for detecting failures at the Orbital Replacement Unit (ORU) level. The generation of rules or algorithms from our models is to be implemented on two separate levels of reasoning. Level 1 is to use qualitative reasoning based on the relationship of how a model's ORU components are connected. Level 2 is to use quantitative measures generated by the modeling of the physics of the system at the ORU level. Simulators for each ORU will be used at this level if they exist.

MEPS is the first of our set of modeling projects. MEPS is to be developed over a number of phases, with the ORU components of an electrical system being represented by icons and the physics of the system being encoded into rules and algorithms. Electrical components for which icons and modeling capabilities will exist are sensors, power sources, breakers/fuses, switches and loads at the ORU level. Additionally the ability to introduce a failure into the model will be provided. Later phases of MEPS call for the modeling of mechanical components which regulate flows and pressures and the sensors that measure these levels. To generate algorithms for the detection of failures, sensors (e.g. voltage, current, pressure, flow rate, status) must be present in the model.

FISCAL YEAR 1990 (continued)

- Alternate Language Interface (ALI)
  - Alarms Triggers & Algorithms (ATA)
    Software Architecture
  - A Graphical Comp Builder (AGCOMPL)
    Graphical Language for Designing Comps
    Language Design derived from Circuit Diagrams
    Graphical Language meant for non-programmers
  - Modeling Electrical Power Systems (MEPS)
    Model Electrical Subsystems/Components
    Model Electrical Component Failures
    Generate Failure Detection Algorithms Using
    Qualitative Reasoning and Quantitative Measures
  - Model Mechanical Components (MMC)
    Model Selected Mechanical Components
    Introduce Mechanical Components into MEPS



RPC : REMOTE POWER CONTROLLER

SW : SWITCH

GMM

Generic Message Management (GMM) is the final project in the ARMOA task. GMM is being designed to provide various message management capabilities for advisory messages sent to a user. While the design of GMM is intended for use with real-time monitoring systems, any system which would require message management could benefit from GMM. There are three functional units to GMM:

- Display Management (DM) - Will manage the screen resources and filter messages using qualitative and time dependency priorities.
- Review Management (RM) - Will manage the reviewing of previous messages and the reviewing of user defined message hierarchies.
- Erase Management (EM) - Will manage the removal of messages from a user defined message hierarchy as well as logging of removed messages for later analysis.

FISCAL YEAR 1990 (continued)

••Generic Message Management (GMM)

•••Display Management (DM)
  Qualitative and Time Dependent Priorities
  Filter Messages
  Manage Screen Resources

•••Review Management (RM)
  Review Previous Messages
  Review Dependent Message Hierarchies

•••Erase Management (EM)
  Log Message to Disk as Removed
  Remove Messages from Internal Hierarchies

INTELLIGENT COMPUTER-AIDED TRAINING

Robert T. Savely
Project Manager
NASA/Johnson Space Center
Software Technology Branch/FR5
Houston, TX 77058

R. Bowen Loftin
Principal Investigator
University of Houston-Downtown
One Main Street
Houston, TX 77002

Intelligent Computer-Aided Training

Participants:

- ROBERT T. SAVELY, NASA/JSC
  TECHNICAL MANAGER

- DR. R. BOWEN LOFTIN, UNIVERSITY OF HOUSTON-DOWNTOWN
  PRINCIPAL INVESTIGATOR

- NASA/JSC SOFTWARE TECHNOLOGY BRANCH
  - CHRIS CULBERT
  - LUI WANG
  - GARY RILEY
  - JAMES VILLARREAL
  - BRIAN DONNELL
  - BEBE LY

- DR. DEBRA S. JOHNSON, UNIVERSITY OF HOUSTON

- COMPUTER SCIENCES CORPORATION
  STEVE BAUDENDISTEL, GRACE HUA, LAC NGUYEN,
  STEVE MUELLER, BOB WAY, TIM SAITO, NISHAD MEHTA,
  ANDRES CALLEGARI, MARCELLO GALARDO,
  REZA RAZAVIPOUR, MARK HUTCHINSON, CHUCK PATTON
  JOE REISEL, JEN WANG

Intelligent Computer-Aided Training

KEY EXTERNAL CONTACTS:

TOM DAVIS, ASTRID HEARD	KSC/PT-TPO
MICHELLE PERRIN	MSFC/EL12
WALT TRUSZKOWSKI	GSFC/522.1
GREGG SWIETEK	NASA-HQ/ST
CHUCK HOLLIMAN	NASA-HQ/MD
FRANK HUGHES, BARBARA PEARSON	JSC/DS2
GEORGE STEPHAN	JPL/601-110

Introduction

Background

- Extensive research in the application of Artificial Intelligence technology to the tutoring task has been performed during the last fifteen years.
- No consensus on the detailed design of an architecture for these systems has emerged from this body of research.
- Significant evidence exists to support the efficacy of intelligent tutoring/training systems.
- The Space Station Freedom Program will require the development and maintenance of large number of training systems.
- Intelligent Computer-Aided Training (ICAT) systems offer the ability to deliver individualized training to large numbers of personnel in a workstation environment.

Introduction

Objectives

- A general architecture for ICAT systems for a wide range of NASA training tasks.
- A suite of software tools to facilitate the development of specific ICAT applications based on the general architecture.

Introduction

Approach

- The general ICAT architecture is being extended and refined through its use in building ICAT systems for use at JSC, KSC, MSFC, and GSFC.
- Existing software tools for use in the ICAT development environment are being evaluated.
- Based on these evaluations and ICAT application development experience, tool requirements are being developed.
- Software tools will be developed and integrated into a comprehensive workstation-based development environment.
- Individual elements of the development environment will be tested in on-going ICAT application projects.
- The integrated development environment will be tested at selected field sites.

Results

ICAT Architecture

The general ICAT architecture consists of

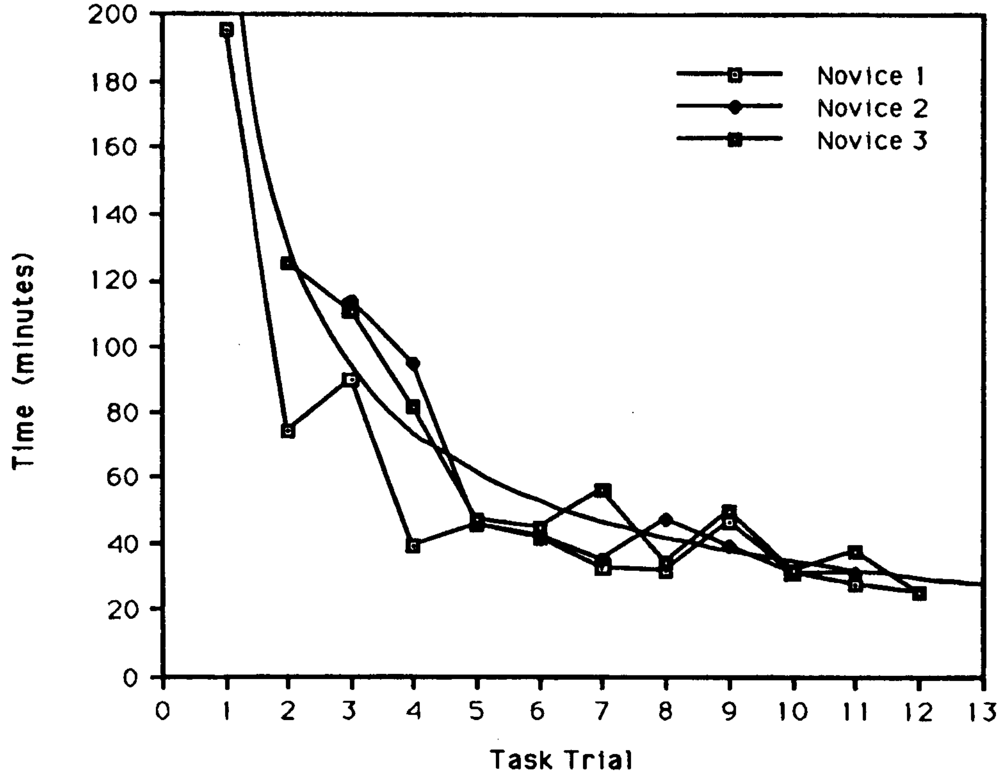
- Four cooperating expert systems
  - Three systems are generic and are applicable to any procedural task
  - The knowledge base related to the task to be trained is built on generic rule templates
- Communication via a common blackboard
- A graphical user interface that duplicates essential components of the task environment and provides an intuitive means for trainee interaction with the system
- An object-oriented database that contains the elements required for training scenario generation
- A dynamic data structure for modeling the trainee's current performance level and past history of system use
- System adaptability to individual trainees' backgrounds and learning rates
- Performance report generation for the trainee and trainer
- Capability of integrating speech recognition and speech generation

Results

ICAT Applications and Projects

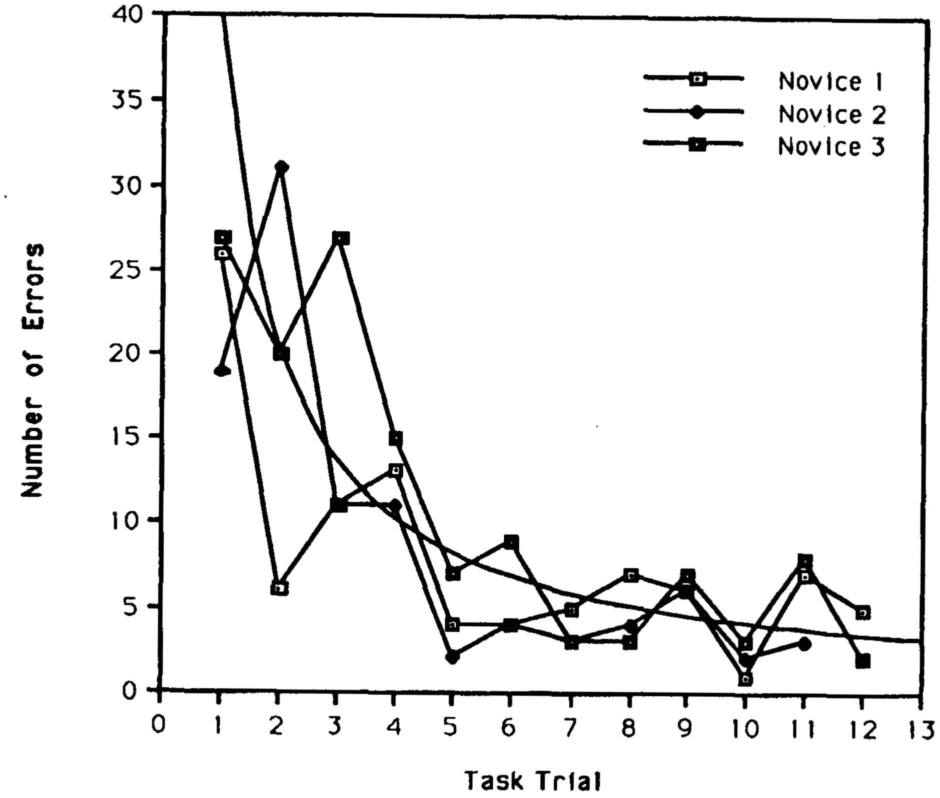
- Payload-Assist Module Deploys/ICAT (PD/ICAT) has been operational and has been used by both novice and experienced Flight Dynamics Officers.
- Vacuum-Vent-Line/ICAT (VVL/ICAT) has been developed and delivered on a PC.
- Main Propulsion Pneumatics/ICAT (MPP/ICAT) is under development for use by test engineers at KSC.
- Instrument Pointing System/ICAT (IPS/ICAT) is under development for Payload and Mission Specialists at MSFC and JSC
- GSFC has initiated the development of an ICAT system for satellite controllers based on the general ICAT architecture.
- SBIR Phase I and II for integration of ICAT technology with existing simulators
- SBIR Phase I for passive knowledge acquisition
- CLIPS Intelligent Tutor
- Propulsion Console Trainer (Air Force)
- Technology Utilization Projects

Results



PERFORMANCE DATA FOR PD/ICAT: TASK COMPLETION TIME

Results



PERFORMANCE DATA FOR PD/ICAT: TRAINEE ERRORS

Results

Knowledge Acquisition Tools

- Twenty-one knowledge acquisition tools have been evaluated.
- Evaluation reports have been widely shared with other NASA centers and key external researchers.
- Two types of knowledge acquisition tools will be required for ICAT development:
  - A tool for capturing knowledge of how to perform a time/event-driven procedural task.
  - A tool for capturing knowledge of the elements required to build training scenarios and trainers' expertise is evaluating trainee performance.
- Detailed requirements for these tools are nearing completion.

Results

<u>User Interface Development</u>

- Extensive investigation of interface development alternatives has led to the selections of X-Windows as the medium of choice.
- Essential common elements of ICAT interfaces have been developed in X-Windows and demonstrated on the Sun, Apollo, Masscomp, and PC. Examples include:
  - Formatted data displays
  - Hierarchical menus
  - Keypads
  - Digitized images
- Development of requirements for a high-level tool for the production of ICAT interface components in X-Windows is underway.

Results

Additional Requirements

Additional requirements are under development for the following software tools:

- Knowledge Base Editor

  An editor is required to aid in adapting the generic rule-bases of the ICAT architecture for specific training applications.

- Object-Oriented Database Editor

  A tool is needed to facilitate the input of scenario definition data into the object-oriented database that creates training scenarios.

- Speech Recognition/Generation

  Tools are required to build the grammars needed by the Speech Systems, Inc. speech recognition system and to assist the linkage of speech recognition/generation to the basic ICAT elements.

Transition

- Demonstrations of ICAT applications for Space Shuttle tasks are underway.
- Cooperation with training organizations with Space Station responsibilities will continue.

Issues

<u>Issues and Problems:</u>

- Integration of ICAT systems with existing and future simulation-based training systems
- On-board Training
  - Long-duration missions
  - Complex tasks infrequently performed
  - Astronaut perceptions
  - Hardware requirements

Conclusions

- A robust and general ICAT architecture is in place and undergoing testing and refinement through application development.
- The efficacy of ICAT systems has been demonstrated through the use of the PD/ICAT system.
- Significant progress has been made in identifying the requirements for tools required to aid in the rapid development of specific ICAT applications based on the general ICAT architecture.

PLATFORM MANAGEMENT SYSTEM EVOLUTION

Michael S. Tilley

Omitron, Inc.

6305 Ivy Lane, Suite 500

Greenbelt, Md 20770

(301) 474-1700

INTRODUCTION

This presentation describes a series of studies, performed in fiscal years 1988 and 1989, to derive and refine a concept of evolutionary growth in capability for the PMS.

The presentation is divided into 6 sections:

The first section, PMS History and Background, describes the origin of PMS as an outgrowth of the manned base OMS, describes the differences between platforms and the manned base which led to the necessity for PMS, and shows the distributed nature of PMS within its ground- and space-based context.

The second section describes the seven basic PMS functions and their interactions.

The third section discusses the history of the PMS evolution process, and shows the series of documents which define the baseline PMS and describe its potential evolution paths.

The fourth and fifth sections respectively, discuss the last two documents in the series completed in fiscal year 1989, the Platform Management System Evolution Plan, and the Platform Management System Design Impacts Report.

The sixth section, Conclusions, summarizes the conclusions reached in the above-mentioned documents.

INTRODUCTION

- PRESENTATION CONTENT:
  - PMS HISTORY AND BACKGROUND
  - THE 7 BASIC PMS FUNCTIONS
  - THE PMS EVOLUTION PROCESS
  - THE PLATFORM MANAGEMENT SYSTEM EVOLUTION PLAN
  - THE PLATFORM MANAGEMENT SYSTEM DESIGN IMPACTS REPORT
  - CONCLUSIONS

PMS HISTORY AND BACKGROUND

The need for a system to perform operations management functions at the base or platform level (as opposed to the system or payload level) was identified in early 1986. The Operations Management System (OMS) was created to encompass these functions. The platforms are out of ground contact more than the base, and as a consequence, require the platform's operation management function to be more autonomous in order to meet the mission goals of the payloads and platform more effectively. The PMS was split from the OMS in mid-1986 and the OMS was redefined to apply only to the base. The PMS was defined in the latter half of 1986 through a series of meetings attended by people from a wide range of organizations involved with the platform. Operations management functions have typically been performed on the ground and have typically been operator intensive. The functions allocated to the PMS are automated to reduce life cycle costs and are split between space and ground. Three criteria were used to assign functions to the PMS:

a) Operations management functions that are required to be on-board;
b) Operations management functions that have the potential to be performed on-board as the platform grows and evolves;
c) Operations management functions that are directly related to items a and b.

The PMS was defined as the system which coordinates the operation of platform systems and payloads. It provides operations monitoring and control and initiates and maintains autonomous platform operations as required. An important PMS concept is that it is only responsible for operations that occur between the systems and payloads. It is not responsible for operations within a payload or system unless they affect the platform's safety or interfere with another scheduled operation.

The level of automation of the PMS functions is expected to increase over the life cycle of the PMS. Because of limited computing resources on the platforms, and because of limited confidence in the automation of operations management functions, only the essential operations management functions that are required for initial operation, and the stubs for future growth, will be implemented on-board the platform at first launch. This subset of the PMS is called the Platform Management Application (PMA) and resides on the Data Management System's (DMS) computer resources. The balance of the PMS is implemented on the ground in the Platform Support Center (PSC). This ground subset is called the Platform Management Ground Application (PMGA). The PMS will be implemented to allow the smooth migration of functions from the PMGA to the PMA.

Figure 1 shows a block diagram of the distributed PMS.

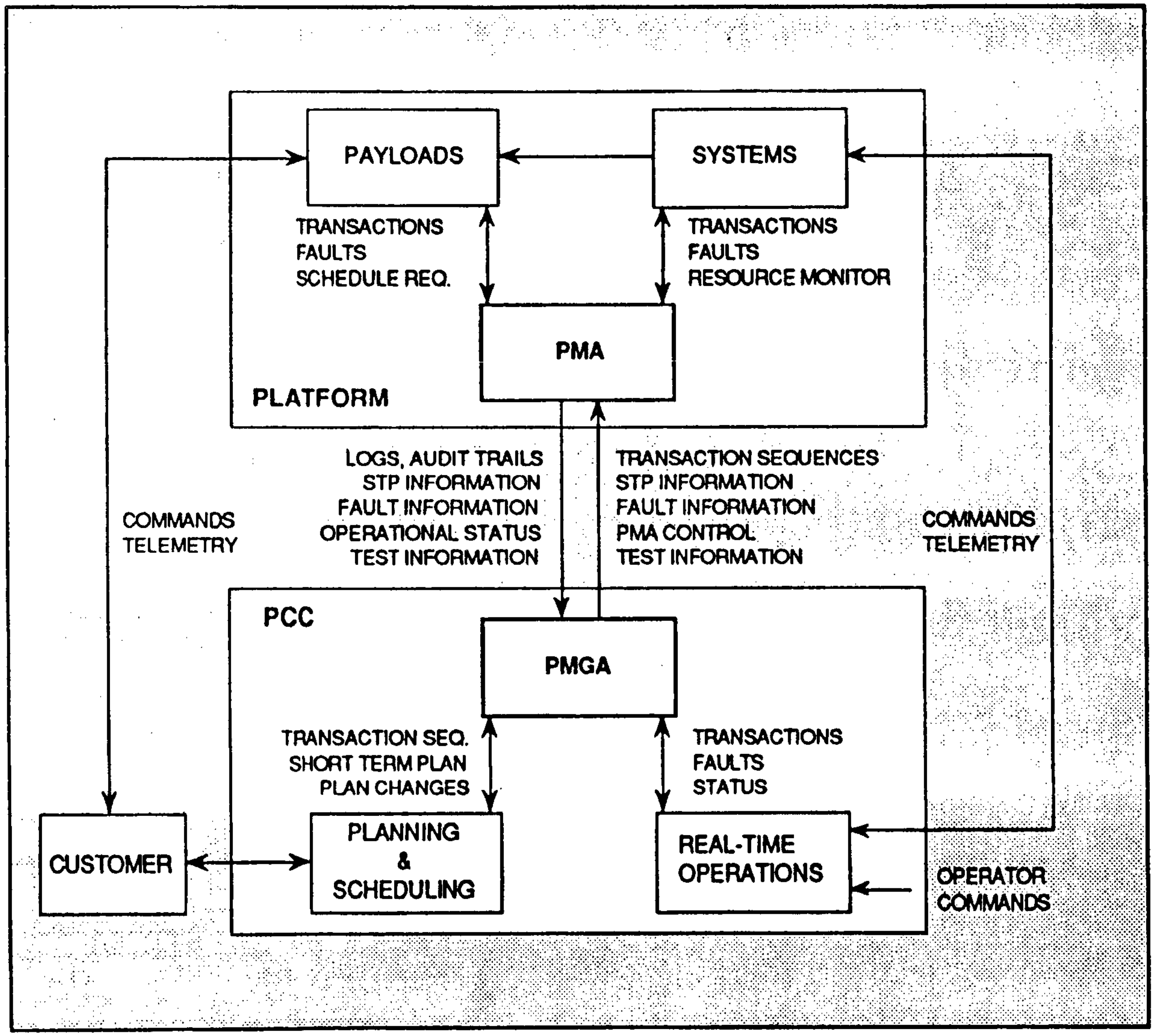


FIGURE 1 PMS DISTRIBUTION

BASIC PMS FUNCTIONS

During the initial definition of PMS, seven basic functions were derived and refined:

Manage the Short Term Plan: The PMS receives a conflict-free Short Term Plan (STP) from the PSC approximately four days ahead of time. The short term plan defines payload and system operation in terms of environmental constraints and resources. The PMS modifies the plan if resource availability changes due to a fault or an incorrect resource prediction. Customers and payloads may request plan modifications (including modifications to ongoing operations). These requests are granted if the resources are available to support the request, or if the priority of the payload and its operation are high enough to displace another operation from the plan.

Manage Operations: The PMS executes the short term plan. It instructs the platform systems on the resources to be supplied to payloads, and controls interlocks on any potentially dangerous payload effectors.

The PMS also provides an optional stored transaction service for customers and operators. Sequences of transactions can be provided to the PMS for execution by time or for execution on the receipt of another transaction. Some transaction sequences may be predefined for use in contingency situations.

Monitor Operations: The PMS receives data from the payloads and systems on resource usage and identifies anomalies. The PMS maintains an audit trail of PMS-related transactions.

Support Testing: The PMS supports on-board interface testing between payloads and/or systems by generating data and logging results.

Recognize and Resolve Conflicts: The PMS recognizes conflicts among the systems and payloads by comparing the resources allocated in the plan and the actual resource availability. The conflict recognition capabilities will be used by the Manage Short Term Plan function in maintaining a conflict-free short term plan. Recognized conflicts will be resolved through a predefined priority scheme.

PLATFORM MANAGEMENT SYSTEM EVOLUTION

Michael S. Tilley
Omitron, Inc.
6305 Ivy Lane, Suite 500
Greenbelt, Md 20770
(301) 474-1700

ABSTRACT

In fiscal year 1988 a study was initiated to define the Platform Management System (PMS) functions required for the mature platform operations era. The objectives of the task include: 1) defining how to increase the operational productivity of the platform by providing enhanced capability for responding to changing events, 2) influencing the initial PMS design by identifying required hooks and scars and, 3) evaluating potential automation techniques that are appropriate given predicted on-board computing resources.

Initial platform operations scenarios were defined. The focus was on PMS-related functions where operations enhancements are likely to occur. Operations productivity was defined in terms of scientific productivity of the platform as well as the level of automation of the ground system. The Platform Operations Productivity Enhancement Report was completed earlier this year documenting system enhancements to increase science productivity and ground system automation.

Using the baseline PMS defined in the PMS Definition Document as a starting point, the resulting PMS-specific enhancements were molded into a sequence of progressively more sophisticated operations management capabilities. This sequence of upgrades to the PMS has been documented in a PMS Evolution Plan. The plan includes enhancements in the areas of resource scheduling, resource modeling, system and payload anomaly management, and transaction sequence interpretation. A plan for migration of functions from the ground portion of the PMS to the flight portion is also included. The impacts of this plan on the platform are now being documented to ensure that the required hooks and scars are included in the baseline system.

Future plans include a prototype of some of the PMS enhancements to address the feasibility of and techniques for implementing these enhancements in the on-board computing environment.

BASIC PMS FUNCTIONS

- MANAGE THE SHORT TERM PLAN
- MANAGE OPERATIONS
- MONITOR OPERATIONS
- SUPPORT TESTING
- RECOGNIZE AND RESOLVE CONFLICTS
- MANAGE INTER-SYSTEM AND PAYLOAD FAULTS
- CHECK TRANSACTIONS

BASIC PMS FUNCTIONS (continued)

Manage Inter-System and Payload Faults: The PMS supervises inter-system and payload fault management and reconfiguration. For certain predefined fault situations, it will support identification of the cause of anomalies and will select an integrated reconfiguration when necessary. If the fault affects the ability of the systems to support the short term plan, the PMS will update the short term plan.

Check Transactions: The PMS will be capable of checking transactions to selected destinations or from selected sources (e.g., PSC operator). Those that are not consistent with the short term plan or safe operation of the platform will not be delivered. The PSC operator can override the check of PSC operator originated transactions.

Figure 2 shows the seven major functions and a greatly simplified view of their interactions.

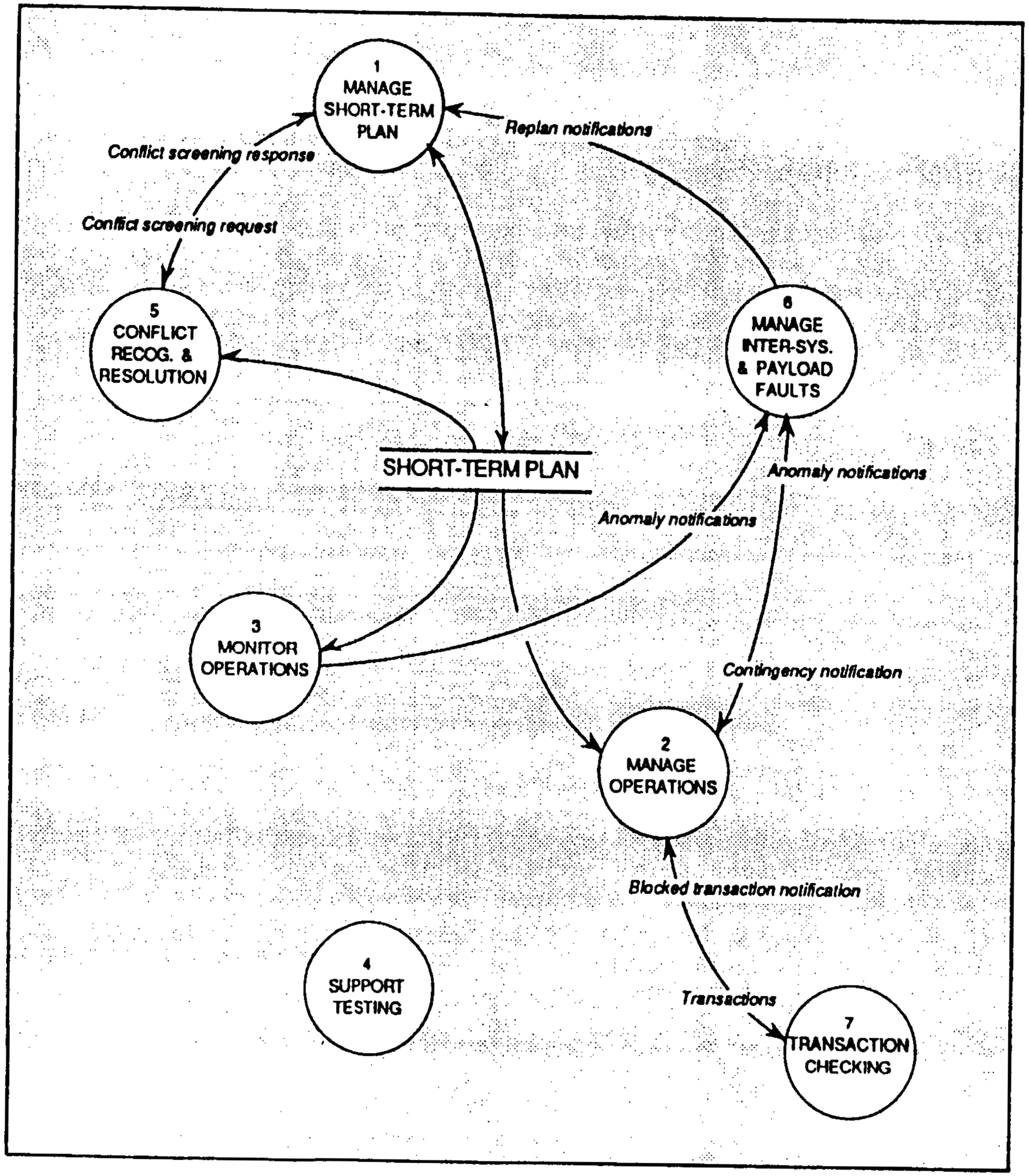


FIGURE 2

THE SEVEN BASIC

PMS FUNCTIONS

THE PMS EVOLUTION PROCESS

The series of 4 documents describing the baseline and enhanced PMS are described below, and illustrated in Figure 3.:

PMS Definition Document (October, 1987)

This document described the PMS functions and interfaces. It allocated functions to the space component and to the ground component. This allocation was for the initial platform launch - other ground functions were expected to migrate to the space component as the platforms grow and evolve. The document also contained a set of dataflow diagrams and a data dictionary.

Platform Operations Productivity Enhancement Report (December, 1988)

The PMS described in the PMS Definition Document was a deliberate compromise. On the one hand, it had to support some fairly radical departures (telescience, reactive control) from traditional spacecraft control methods. On the other hand, it had to be kept as simple as possible in recognition of the following:

- the first flight PMS will probably suffer from a severe shortage of computational resources
- the first flight PMS should be simple to minimize operational risk
- as on-board computing resources increase (through later generations of platform, or through servicing of existing platforms) PMS capabilities can be enhanced
- as operational experience is gained, PMS capabilities can be enhanced in areas once considered to be high-risk

In 1988, the Platform Operations Productivity Enhancement Report was delivered. Using the existing PMS definition as a departure point, this document examined ways to increase the operational productivity of the platform, e.g., how to provide the capability to react to changing conditions, such as targets of opportunity and faults, in a manner that comes closer to optimizing the overall platform mission operation. This was achieved by analysis of a number of scenarios, some from flight experience of existing traditional spacecraft, others based on expected platform operations. The results were documented as a series of enhancements to the basic PMS functions.

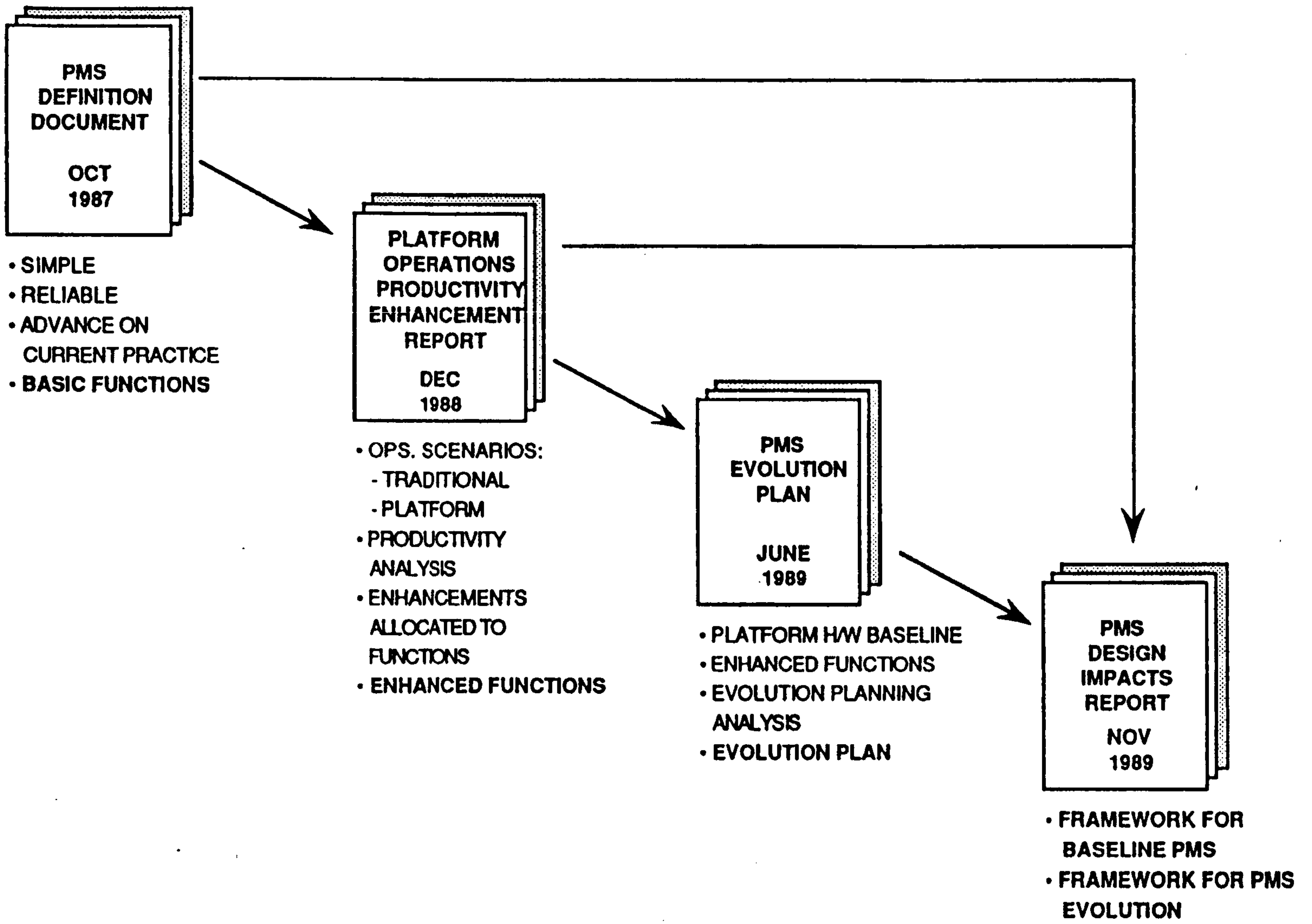


FIGURE 3 PMS EVOLUTION PROCESS

THE PMS EVOLUTION PROCESS (continued)

Platform Management System Evolution Plan (June, 1989)

The purpose of this document was to define a growth path from the baseline Platform Management System to an enhanced PMS based on the Platform Operations Productivity Enhancement Report. The document described the benefits and implementation feasibility of each proposed enhancement, and allocated each enhancement to a PMS evolution phase. It was intended for this document to be used by GSFC Flight Software Systems Branch in monitoring the initial implementation of PMS, and in the maintenance and enhancement of future versions.

Platform Management System Design Impacts Report (to be published November, 1989)

This document, the last in the series, describes the attributes of the Space Station Freedom platform (and its systems) necessary to allow the baseline PMS to function as intended, and notes those attributes of the initial platform which will be necessary to allow PMS evolution. The intended use of the document is to influence the initial design of platform systems to ensure that the infrastructure will exist to allow PMS to fulfill its intended role, and to influence the initial design of PMS itself to ensure that PMS evolution can be achieved at acceptable cost and risk.

PLATFORM MANAGEMENT SYSTEM EVOLUTION PLAN

In considering PMS evolution it is necessary to assume a life-cycle model for the platform hardware and software. While the platform software can be incrementally updated (within the limits of available computing resources), the platform hardware presents only a few discrete opportunities for change - either during on-orbit servicing, or during development of a new platform. Recently, the concept of servicing for polar platforms has been de-emphasized, so that there will be no hardware upgrade during the life of an individual polar platform. However, as new, expendable polar platforms are developed, evolved hardware and software concepts can be applied to them. For co-orbiting platforms, servicing options are still available . Of course, the PMS concept is intended to be applicable to both co-orbiting and polar platforms.

As a device to break up the platform life-cycle into more manageable units, bearing in mind the above discussion, it seems reasonable to suppose the following attributes of a platform life-cycle model:

- first upgrade five years after first launch (either as a result of servicing, or a new platform - current Eos planning involves launching new platforms at 5-year intervals),
- limited hardware changes (except for changes to the payload complement) at the first upgrade,
- second upgrade ten years after first launch,
- potential for extensive hardware change at the second upgrade, for example:

    - replacement of computational resources (standard data processors (SDPs), etc.) with faster devices,
    - addition of more SDPs,
    - additional mass storage.

This model allows three major evolutionary phases for PMS (illustrated in Figure 4):

• BASELINE PMS	from launch to first upgrade,
• INTERMEDIATE PMS	from first upgrade to second upgrade,
• MATURE PMS	after second upgrade.

Although useful for planning purposes, the boundary between the phases is somewhat artificial. Flight software can be updated at any time in the life-cycle provided that on-board resources are available.

At any given time, it quite likely that there will be several platforms in operation at different points in this life-cycle.

The development of hardware and software ground components of the PMS, the Platform Management Ground Application (PMGA), is not constrained to the same degree as the flight hardware and software. However, it is likely that major changes to PMA and PMGA will occur in concert, so it is convenient to use the same life-cycle phases for both.

PLATFORM MANAGEMENT SYSTEM EVOLUTION PLAN

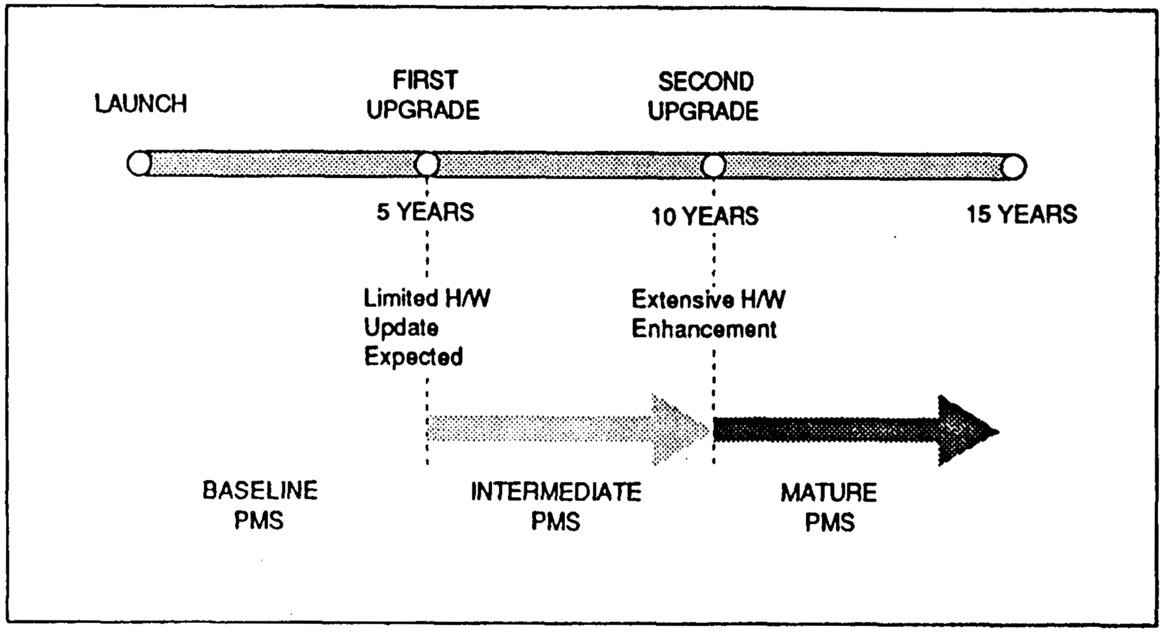


FIGURE 4 PMS EVOLUTION PHASES

PMS ENHANCEMENTS

The PMS Evolution Plan analyzed a number of enhancements proposed in the Platform Operations Productivity Report. These are briefly described below. Each enhancement was allocated a mnemonic identifier (EH-1, etc.) for reference:

EH-1 OBSSTP Integrity Check: Filter the uplinked Short Term Plans (STPs) for elementary errors such as inappropriate time-spans or payload identifiers. The uplinking of incorrect command loads has been a frequent problem on past spacecraft. This is also addressed in EH-5, below. In the PMS era, some of the functions of command loads in traditional spacecraft are embodied in the STP, and so a second aspect of the same problem is the transmission of STP elements and associated transactions to PMS. In this area again, elementary checks performed by PMS could eliminate many problems.

EH-2 Multiple Resource Profiles: Compare the On-Board Segment of the STP (OBSSTP) against several possible resource profiles and choose the best fit, when there is a change in resource availability and a corresponding choice of reconfiguration options. The reconfiguration options are assumed to be provided by the subsystems. A typical situation would be a partial failure of a resource-providing subsystem. Even when autonomous recovery is effected by the failing subsystem, where there is a choice of recovery options, PMA will still be involved. For example, suppose that the Electrical Power System (EPS) has suffered a partial failure and has two alternative recovery actions available - both will result in a reduced power availability profile but, as far as the EPS is concerned, both profiles are equally "good". Clearly PMA will have to determine which profile provides the best fit (i.e., least disruption) to the OBSSTP.

EH-3 OBSSTP trade-offs: Determine the optimum configuration for current OBSSTP when more complex trade-offs are possible. For example, in the case described above, a more favorable (i.e. more productive) power profile might be obtained by changing the platform attitude to gain better solar array coverage, at the expense of some loss of FOV for some instruments. This type of trade-off requires enhancements to several functions. However, recognizing the possibility of alternative strategies (the "what if"s) requires an entirely different type of processing ("Artificial Intelligence" for lack of a better term).

PMS ENHANCEMENTS

- OBSSTP INTEGRITY CHECKS (EH-1)
- MULTIPLE RESOURCE PROFILES (EH-2)
- OBSSTP TRADE-OFFS (EH-3)

PMS ENHANCEMENTS (continued)

EH-4 Transaction Interpreter: Implement a transaction interpreter, which would allow a transaction sequence to examine operational conditions, and change execution flow accordingly. This enhancement first surfaced as a possible approach to the provision of a flexible response to faults. The fault recovery of the baseline PMA consists of a combination of hard-coded algorithms and contingency transaction sequences. The nature of a transaction sequence has not been defined in any depth, but seems to be equivalent to traditional spacecraft commands in most respects. Clearly, a predefined command sequence of this type embodies almost no flexibility at execution time. If, however, transaction sequences were expressed in a command language with the ability to interrogate the environment (retrieve engineering values) and execute conditional statements, a great deal more flexibility would be available. This type of approach is used to automate functions in control centers and integration & test systems today. A typical example is the Systems Test and Operations Language (STOL).

EH-5 Integrity Check on Transaction Sequences: Perform basic integrity checks on uplinked transaction sequences. Uplinking of incorrect command loads has been particularly troublesome on past missions. In the PMS era, the baseline PMS can eliminate most of the problems normally caused by human carelessness (it will not forget to send a sequence at the required time; it will not send a sequence with a similar name to the sequence actually required; etc.). The kind of checking required to perform elementary validation of transaction sequences (which should not be confused with Transaction Checking) is not very complicated.

EH-6 OBSSTP screening: When there is a change in resource availability, and a corresponding choice of reconfiguration options, this function will be instrumental in comparing the OBSSTP against several possible resource profiles and choosing the best fit. Note that this enhancement is closely tied to EH-2 The division of labor is as follows:

- EH-2 recognizes the availability of several recovery options,
- EH-10 provides resource availability profiles for each option,
- EH-6 determines which profile has the best fit with the OBSSTP,
- EH-2 chooses the corresponding recovery option.

This enhancement is so closely connected with EH-2 that it is not useful to consider them separately as stand-alone enhancements (they have been described separately up to this point only because of the way they were derived in Reference [3]). For the remainder of this document, we will combine EH-2 and EH-6 into EH-12 , Multiple Resource Profiles, and discuss the combination as a single logical enhancement.

PMS ENHANCEMENTS (continued)

- TRANSACTION INTERPRETER (EH-4)
- INTEGRITY CHECK ON TRANSACTION SEQUENCES (EH-5)
- OBSSTP SCREENING (EH-6)

PMS ENHANCEMENTS (continued)

EH-7 Improved Schedule/Reschedule Algorithms: The baseline PMS is handicapped in dealing with many problems because of the lack of scheduling information in the STP. In the event of a schedule conflict, PMA's only recourse is to progressively delete lower priority plan elements until the conflict disappears. This, in itself, will often be a sub-optimal solution. There are two distinct ways in which the situation can be improved. The first involves enhancements to baseline functions to allow plan elements to be moved in time. For example, in the event of an unexpected excess of resources, if the OBSSTP contained the necessary information an evolved PMA could move plan elements forward in time to make use of the available resources. This would not, in itself, improve productivity - the same elements would be executed but at different times. It would, however, improve scheduling flexibility by freeing resources later in the plan. These could then be allocated to payloads making real-time requests, or used in other ways as suggested below. The ability to move elements is also useful in other circumstances. For example, in the event of a resource conflict it might be possible to rearrange the STP elements so that more of them could execute than under the simple deletion scheme of the baseline PMA. This level of enhancement requires changes to PMS to add some genuine rescheduling ability. In addition, considerable information would have to be included in the OBSSTP relating to the scheduling requirements and constraints of each plan element. For example:

- schedule this element only between these times ...
- do not schedule this element while payload X is operating
- this element and element n should be considered as a unit and scheduled together
- do not start this element unless at least x minutes of ground contact remain before the next communication outage.

The second way in which PMA scheduling could be improved is discussed in EH-8, below.

EH-8 Insertion of Unscheduled Events: Support insertion of unscheduled events into the plan. An additional enhancement to PMA's scheduling ability relates to observations which are not contained in OBSSTP at all. In some cases there may be elements of an experimenter's schedule request which were not included in the STP because of resource conflicts or other constraints. If PMA had available a pool of such unscheduled elements, an evolved scheduler could insert them into the plan if the constraints relaxed for any reason. It is also possible that some opportunistic instruments could be routinely operated in this way - scheduled on a "fit this in when you can" basis.

PMS ENHANCEMENTS (continued)

EH-9 Intelligent Anomaly Diagnosis: Intelligent fault recovery requires fault diagnosis and isolation down to the level at which recovery options can be selected, to provide the potential recovery strategies for Conflict Recognition and Resolution to evaluate. This type of goal-directed processing is more usually associated with Artificial Intelligence and Expert System approaches and is difficult to insert into traditionally structured systems. In order to accommodate this enhancement, a different PMS infrastructure might be indicated.

EH-10 Resource Availability Prediction: Maintain (and possibly derive) resource availability predictions. The resource availability profile is of great importance in the recovery from faults, and in scheduling in general. To use a power example again, assume that the EPS has suffered a partial failure and has a choice of two (or more) reconfiguration options. The two choices may result in quite different power availability profiles over the remainder of the orbit and beyond. By itself, the EPS has no way of evaluating the relative merits of these profiles - only PMS can do that with reference to the resource needs of the scheduled operations. Given the availability of predicted resource profiles, PMA could test its OBSSTP against them and use the priorities embedded in the OBSSTP to select the resource profile having the best fit. This function could be relatively trivial (if the subsystems were able to provide their own resource predictions) or quite complex (if it had to model the behavior of one or more subsystems).

EH-11 Manage Payload Anomaly: If we assume that some of the proposed enhancements to PMS involve the use of AI technology, then this implies the existence of an on-board inference-engine of some type and a body of expert knowledge relating to the platform and its subsystems. This body of knowledge could be a set of rules, for example. Under these circumstances, it is conceivable that a set of rules could be supplied for a particular payload. These rules, in conjunction with PMA's inference engine, could act as an "expert advisor" in handling anomalies of the payload. This approach allows PMS a degree of "smartness" about the payload and its activities without compromising its "purity". The storage and use of payload-specific rule sets could be considered to be a PMA service in much the same way as transaction sequences are managed by the baseline PMA.

EH-12 Multiple Resource Profiles: As explained above, EH-12 is a combination of EH-2 and EH-6.

PMS ENHANCEMENTS (continued)

- IMPROVED SCHEDULE / RESCHEDULE ALGORITHMS (EH-7)
- INSERTION OF UNSCHEDULED EVENTS (EH-8)

PMS ENHANCEMENTS (continued)

- INTELLIGENT ANOMALY DIAGNOSIS (EH-9)
- RESOURCE AVAILABILITY PREDICTION (EH-10)
- MANAGE PAYLOAD ANOMALY (EH-11)
- MULTIPLE RESOURCE PROFILES (EH-12)

EVOLUTION PLAN

For each of the enhancements proposed in the previous section, the following topics were examined:

- the benefits of the enhancement to platform and payload operations,
- the feasibility of implementing the enhancement, in terms of:
    - availability of the necessary platform resources,
    - availability of the necessary technology,
- allocation of the enhancement to one of the PMS evolution phases discussed above:

- PMA/PMGA baseline	(can and should be added to baseline),
- PMA/PMGA intermediate	(1st major s/w upgrade, h/w unchanged),
- PMA/PMGA mature	(2nd major s/w upgrade, possible h/w upgrade),

In addition to the factors listed above, the allocation to PMS evolution phases also considered the dependencies between the proposed enhancements, since few of them are completely stand-alone.

Finally, the document summarized the time-phased implementation of the enhancements. These are illustrated in Figures 5 and 6 (the phasing of EH-10 is shown separately in Figure 6 for clarity). The migration of functions from PMGA to PMA is also shown. For the sake of discussion, the time-phasing corresponds to the evolution phases already presented , however, with respect to software changes, there is no reason to artificially restrict updates to predetermined servicing epochs. Software changes that are essentially stand-alone may be implemented or migrated in a continuous fashion, with due consideration for platform resources and ongoing platform operations.

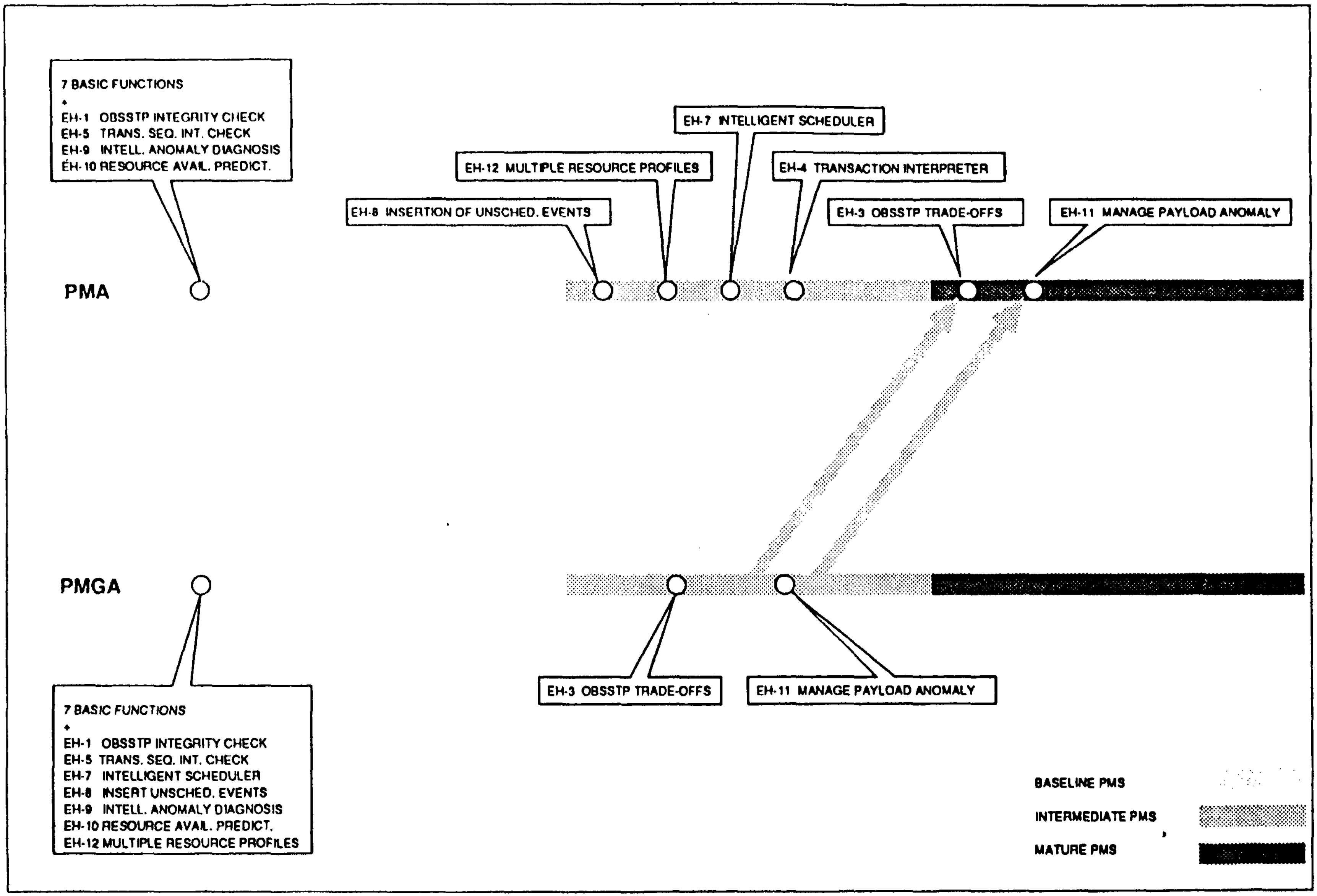


FIGURE 5 PMS EVOLUTION AND MIGRATION PHASING

EVOLUTION PLAN (continued)

Enhancement EH-10 (Resource Availability Prediction) has a wide range of implementation options depending on the capabilities of individual subsystems to predict their own resource availability profiles. To avoid too much clutter on Figure 5, the evolution and migration phasing of EH-10 is shown separately in Figure 6.

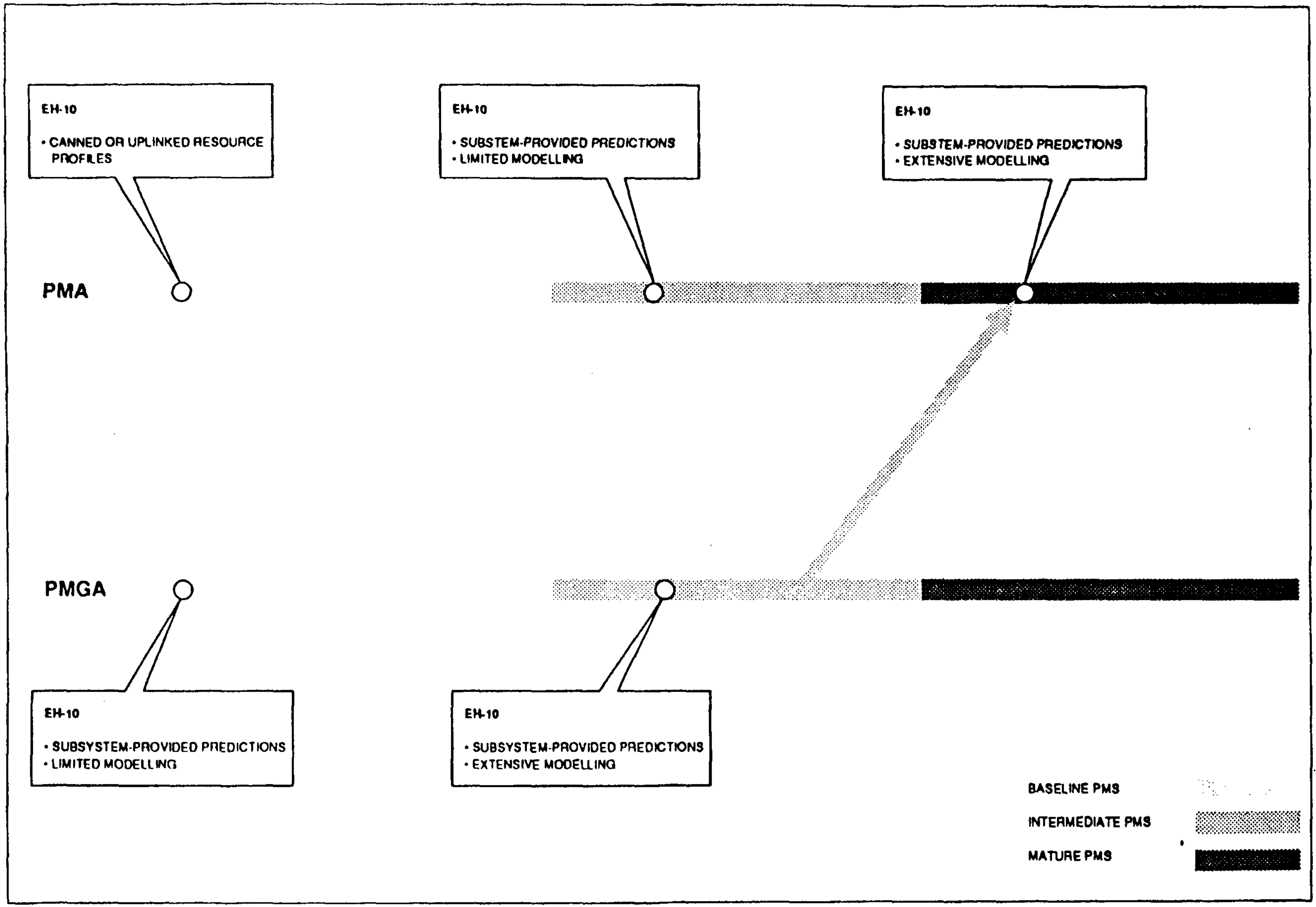


FIGURE 6 EH-10 (RESOURCE AVAILABILITY PREDICTION) EVOLUTION AND MIGRATION PHASING

PLATFORM MANAGEMENT SYSTEM DESIGN IMPACTS REPORT

The PMS Design Impacts Report consists of 3 main sections. The first considers the platform hardware and software attributes required by the baseline PMS. It is divided into the following sub-sections corresponding to the platform resource providers:

- Data Management System (DMS)
- Communication and Tracking (C&T) System
- Electrical Power System (EPS)
- Guidance, Navigation and Control (GN&C) System
- Thermal Control System (TCS)

The second section considers the platform hardware and software attributes required to allow PMS to evolve into the more advanced system described in the PMS Evolution Plan. The opening sub-section discusses the required evolutionary attributes of the baseline PMS itself. The remaining sub-sections are concerned with the major platform systems (as above).

In the previous sections (and also in the preceding documents) the PMS baseline and enhanced functions were discussed in terms of their usefulness, without much concern over how they could be implemented. The third section of the Design Impacts Report examines the potential difficulties in PMS implementation, looks at work in progress to alleviate some of these difficulties, and suggests some additional studies. This section is divided into sub-sections corresponding to a natural grouping of PMS implementation issues into the following areas of implementation risk:

- flight CPU,
- scheduling,
- artificial intelligence,
- subsystem modelling,
- on-board transaction interpreter.

PLATFORM MANAGEMENT SYSTEM DESIGN IMPACTS REPORT

- THREE MAJOR SECTIONS:
  - PLATFORM H/W AND S/W ATTRIBUTES REQUIRED TO IMPLEMENT BASELINE PMS
  - PLATFORM H/W AND S/W ATTRIBUTES REQUIRED FOR PMS EVOLUTION
  - POTENTIAL IMPLEMENTATION PROBLEMS AND SUGGESTED FURTHER WORK

PLATFORM MANAGEMENT SYSTEM DESIGN IMPACTS REPORT - BASELINE H/W & S/W REQUIREMENTS

The document lists a number of requirements on the platform systems which are required for the implementation of the baseline PMS. It is not useful to reproduce the requirements here, however, the document lists requirements in the following areas:

- on-board CPU
- local area networks
- database and file management
- tape-recorders
- guidance, navigation and control
- system services
- mass storage
- downlink
- electrical power
- thermal control

In addition to the specific requirements as mentioned above, there are several general requirements applicable to all subsystems:

- each subsystem should be able to measure and control the resource(s) it provides
- each subsystem must allow PMS to specify the level of its resource to be supplied to each payload
- each subsystem should support fault isolation and recovery

Finally, several requirements for on-board tape recorder (T/R) control are of particular note:

- In order for PMA to monitor usage of the tape recorder resource, PMA requires the T/R system to:
    - measure incoming bit rates from each payload
    - report these usage rates to the PMA

- In order to allow rescheduling of resources by the PMA in response to faults or requests from users, the T/R software must have a tape recorder scheduling function which:
    - receives a tape recorder resource envelope at time t1 from the PMA which specifies both the planned ingest rates (in bps) for each of the payloads and the downlink bandwidth available at time t2,
    - calculates the best configuration of operating tape recorders, record/playback speeds, and bitstream switching to support the envelope at time t2,
    - generates the appropriate tape recorder and data stream switching commands between time t1 and t2 to effect the new T/R configuration.

PLATFORM MANAGEMENT SYSTEM DESIGN IMPACTS REPORT - BASELINE H/W & S/W REQUIREMENTS

- BASIC REQUIREMENTS FOR:
  - ON-BOARD CPU
  - SYSTEM SERVICES
  - LOCAL AREA NETWORKS
  - MASS STORAGE
  - DATABASE AND FILE MANAGEMENT
  - DOWNLINK
  - TAPE-RECORDERS
  - ELECTRICAL POWER
  - GUIDANCE, NAVIGATION AND CONTROL
  - THERMAL CONTROL

- GENERAL REQUIREMENTS FOR:
  - RESOURCE MEASUREMENT AND CONTROL
  - FAULT ISOLATION AND RECOVERY

PLATFORM MANAGEMENT SYSTEM DESIGN IMPACTS REPORT - EVOLUTION REQUIREMENTS

Very few evolutionary requirements on platform subsystems were identified. As in the case of the baseline PMS, the limits on PMS evolution will ultimately be decided by the on-board computing infrastructure (CPUs, LANs, mass storage, etc.) more than any other factor. Unless there is flexibility and transparent extensibility in these areas, evolution will be difficult or impossible.

Although the platform environment is important in determining the limits of PMS evolution, the baseline PMS itself needs to be implemented in a way that will permit both evolution of functions, and migration of functions from the ground:

- PMS design needs to stress modularity with particular emphasis on defining function boundaries such that, as far as possible, the effects of any evolutionary change are confined to a single function.

- To facilitate function migration from ground to flight segments, PMS design needs to stress transportability. Originally, this was mandated by Space Station Freedom policy (for example, the use of a common, transportable operating system for both flight and ground elements). Although the polar platform is no longer bound by this mandate, the potential benefits should not be discarded hastily.

- Several proposed enhancements require additional information to be present in the STP. Design of the STP should be such that its structure can be changed without requiring code changes to the flight software. Changes to flight code will then only be required if new STP fields are added, or if the meaning of an existing field is changed.

- Several proposed enhancements have a significant Artificial Intelligence (AI) content. Rather than attempt to add these to an existing, traditional program structure, it would be beneficial to consider designing PMS from the outset as a distributed, extensible AI system.

PLATFORM MANAGEMENT SYSTEM DESIGN IMPACTS REPORT - EVOLUTION REQUIREMENTS

- MAJOR EVOLUTION REQUIREMENT ON PLATFORM IS FLEXIBILITY AND EXTENSIBILITY OF COMPUTING INFRASTRUCTURE (CPUs, LANs, MASS STORAGE, ETC.)
- THE BASELINE PMS ITSELF NEEDS TO BE DESIGNED FROM THE OUTSET WITH MIGRATION AND EVOLUTION IN MIND

FLIGHT CPU

As mentioned above, PMS is a software system and, like any software system, is potentially constrained by the CPU on which it executes. In the case of PMS, the situation is complicated by a number of unknown factors:

- the amount of on-board CPU-time available to PMS is unknown,
- the amount of processing that can be done with the available CPU-time (i.e., the speed of the CPU) is undecided,
- the amount of on-board RAM available to PMS is unknown.

Given the original Space Station Freedom (SSF) platform architecture, none of these represented insurmountable problems. The CPU was fast enough and expandable enough that, despite the uncertainties over the available CPU budget, implementation of the baseline PMS seemed very plausible. It was also clear that the budget would be exceeded by the proposed evolutionary enhancements. However, since there was a phased implementation strategy for the PMS enhancements, and a clear growth path for the CPU, the evolutionary enhancements also appeared plausible.

Recently, with the separation of the polar platform from the Space Station Freedom Program, the choice of on-board CPU has again been thrown open. While this creates additional uncertainties, there are two ways that the situation could be improved:

- The PMS functions should be modelled and prototyped to establish realistic CPU and RAM requirements. Some of this has already been undertaken in the PMS testbed for the baseline functions. Further prototyping of the enhancements should be carried out to provide the basis for a realistic implementation plan.

- The choice of the on-board CPU should be made very carefully with regard to an available growth path. This is an issue not just for PMS, but for the evolution of the platform as a whole. The SSF CPU (Intel 386) has a very clear growth path to the 486 (already shipping), the 586 (already produced in small quantities) and beyond. This continuing growth in CPU capability and reduction in power consumption is driven by relentless commercial pressures. The benefits are available to NASA at practically zero cost. The trade-off between these benefits and the problems of radiation hardening (which will probably never be addressed by the commercial sector) must be made very cautiously. An inappropriate choice at this point could restrict a whole generation of platforms.

PLATFORM MANAGEMENT SYSTEM DESIGN IMPACTS REPORT - IMPLEMENTATION ISSUES

- FLIGHT CPU
    - PMS CPU BUDGET
    - FLIGHT CPU PERFORMANCE
    - RAM AVAILABILITY
    - CHOICE OF CPU ON REDESIGNED PLATFORM

PLATFORM MANAGEMENT SYSTEM DESIGN IMPACTS REPORT - IMPLEMENTATION ISSUES

(continued)

SCHEDULING

The very simple rescheduling capability of the baseline PMS poses no great implementation challenges. However, several of the proposed enhancements involve more sophisticated methodologies which pose a greater implementation risk. There are two distinct challenges. The first is that 'smart' schedulers tend to be resource intensive which, in turn, becomes another facet of the CPU/RAM budget problem discussed above. The second challenge is that of scheduling technology itself. Although a lot of work is being done in this area, the platform environment imposes some unusual constraints:

- 'smart' schedulers have not been tried in autonomous flight systems,
- there is not much experience with schedulers operating in real-time.

These concerns have been at least partially addressed in two ways:

- Computer Sciences Corporation has been working with GSFC Code 520 on an automated scheduling task funded by the SSF Transition Definition Program. This task has implemented a prototype PMGA and PMA scheduler running on an IBM PC computer. This prototype demonstrates real-time rescheduling in response to events, with synchronization of the schedulers taking place during simulated ground contacts. In addition to exploring interactions between the schedulers, the prototype demonstrates real-time conflict recognition and resolution. The schedulers are based on the PMS baseline definition with an additional capability of adding new elements in response to an increase in available resources (a similar capability is one of the proposed PMS enhancements).
- General Electric (GE)Astro Space Division proposed a scheduler for PMS, having many of the enhanced capabilities, based on an existing PC-hosted Landsat scheduler. The current status of this project is unknown.

PLATFORM MANAGEMENT SYSTEM DESIGN IMPACTS REPORT - IMPLEMENTATION ISSUES

(continued)

- SCHEDULING
  - SMART SCHEDULERS
  - SCHEDULING IN REAL-TIME ENVIRONMENT
  - CODE 520 AUTOMATED SCHEDULING TASK
  - GE LANDSAT SCHEDULER

PLATFORM MANAGEMENT SYSTEM DESIGN IMPACTS REPORT - IMPLEMENTATION ISSUES

(continued)

ARTIFICIAL INTELLIGENCE

Several of the proposed PMS enhancements identified a need for problem solving of the type often ascribed to artificial intelligence or expert systems, particularly in the area of anomaly diagnosis. As in the previous section, this poses challenges because of the limited CPU availability, and also because the technology is somewhat unexplored, especially in flight systems.

Another GE proposal for the SSF platform was an Expert System for Platform Anomaly Diagnosis (ESPAD). In addition to meeting the requirements of one of the proposed PMS enhancements (EH-9), it appeared that several other enhancements could be achieved via extensions to the ESPAD rule-base. A prototype of the inference engine exists and has returned promising benchmark results. Once again, the current status of this project is uncertain.

SUBSYSTEM MODELLING

One of the things that emerged from the Platform Operations Productivity Enhancement work is the importance of good resource predictions in enabling an advanced scheduler to make good scheduling decisions. This involves modelling the behavior of subsystems to a reasonable degree of fidelity, and could be done by the subsystem, or by PMS, or a mixture of both. While this has been done in ground engineering systems it is, once again, a novel application for flight systems. This type of modelling is a good candidate for migration from ground to flight systems. It is also possible that significant results could be obtained using a rule-based system such as ESPAD.

ON-BOARD TRANSACTION INTERPRETER

The proposed on-board transaction interpreter is unusual when compared to the other enhancements, in that it does not require large amounts of CPU resources, and the technology is very well understood from extensive ground-system experience. However, this enhancement did arouse some concern over the possible operational impacts. It might be fruitful to prototype the transaction interpreter, not to explore the technology, but to demonstrate the potential operational benefits.

PLATFORM MANAGEMENT SYSTEM DESIGN IMPACTS REPORT - IMPLEMENTATION ISSUES (continued)

- ARTIFICIAL INTELLIGENCE
- SUBSYSTEM MODELLING
- ON-BOARD TRANSACTION INTERPRETER

CONCLUSIONS

The PMS definition and evolution documents described above have demonstrated a number of important conclusions:

- Implementation of a Platform Management System does involve more risk than a traditional spacecraft on-board system, however, the potential benefits are numerous:
    - greater flexibility in platform operations,
    - better response to unplanned events (targets of opportunity),
    - less observation time lost in case of recoverable faults,
    - increased science data return,
    - more efficient use of platform resources
    - reduced costs for ground operations
    - directly supports improved user/instrument interaction (telescience).
- In order to achieve these benefits, certain characteristics of the platform and its subsystems are required:
    - powerful on-board CPU(s),
    - flexible on-board communications (LANs),
    - resource-providing subsystems able to measure and control the resources they provide
    - an architecture that supports incremental and transparent growth in on-board computational capability.

Given the above characteristics, it is possible to implement a platform which is more capable than traditional systems by orders of magnitude, and which can provide a suitable foundation for years of future growth.

ACKNOWLEDGEMENTS

The PMS evolution effort was funded by the Office of Space Station Strategic Plans and Programs Division, and managed by Goddard Space Flight Center Flight Software Systems Branch.

REFERENCES

1. "Space Station Operations Task Force Summary Report," NASA, October, 1987.

2. Fishtahler, L. / Tilley, M./ Tompkins, S., "Platform Management System (PMS) Definition Document," GSFC, October, 1987.

3. Hawkins, F. / Simm, D. / Tilley, M., "Platform Operations Productivity Enhancement Report," GSFC, December, 1988.

4. Stakem, P. / Tilley, M., "Platform Management System Evolution Plan," GSFC, June, 1989.

5. Hartley, J. / Tilley, M., "Platform Management System Design Impacts Report," GSFC, November, 1989.

AUTOMATED
PLATFORM MANAGEMENT SYSTEM
SCHEDULING

Larry G. Hull
Code 522
Goddard Space Flight Center
Greenbelt, Maryland 20771

ABSTRACT

The Platform Management System was established to coordinate the operation of platform systems and instruments. The management functions are split between ground and space components. Since platforms are to be out of contact with the ground more than the manned base, the on-board functions are required to be more autonomous than those of the manned base. Under this concept, automated replanning and rescheduling, including on-board real-time schedule maintenance and schedule repair, are required to effectively and efficiently meet Space Station Freedom mission goals.

In a FY88 study, we developed several promising alternatives for automated platform planning and scheduling. We recommended both a specific alternative and a phased approach to automated platform resource scheduling. Our recommended alternative was based upon use of exactly the same scheduling engine in both ground and space components of the platform management system. Our phased approach recommendation was based upon evolutionary development of the platform.

In the past year, we developed platform scheduler requirements and implemented a rapid prototype of a baseline platform scheduler. Presently we are rehosting this platform scheduler rapid prototype and integrating the scheduler prototype into two Goddard Space Flight Center testbeds, as the ground scheduler in the Scheduling Concepts, Architectures, and Networks Testbed and as the on-board scheduler in the Platform Management System Testbed. Using these testbeds, we will investigate rescheduling issues, evaluate operational performance and enhance the platform scheduler prototype to demonstrate our evolutionary approach to automated platform scheduling.

The work described in this paper was performed prior to Space Station Freedom rephasing, transfer of platform responsibility to Code E, and other recently discussed changes. We neither speculate on these changes nor attempt to predict the impact of the final decisions. As a consequence some of our work and results may be outdated when this paper is published.

INTRODUCTION

The Platform Management System (PMS) has been established to coordinate the operation of platform systems and instruments. The management functions are split between ground and space. The ground segment is designated the Platform Management Ground Application (PMGA). The space segment is the Platform Management Application (PMA). The PMS Definition Document (Reference 1) prescribes that each application includes seven functions. Two of these functions are associated with the job of maintaining a platform resource schedule. The Platform Management System must only alter this resource schedule in response to change requests and changes in resource availabilities.

Schedule generation is not a function allocated to the Platform Management System but rather it is performed by a Platform Support Center scheduler which furnishes a short term plan. The PMS manages the short term plan and performs rescheduling (the PMS conflict recognition and resolution function). Rescheduling is of particular interest because it is initiated from three sources: instrument, end user, and the platform itself. As shown in Figure 1, there are three schedulers of different capabilities involved.

o An on-board scheduler is part of the PMA. Initially, the on-board scheduler will only reschedule to the extent necessary to ensure platform and instrument safety until the next contact.

o A ground scheduler is part of the PMGA. This scheduler is more capable than the on-board scheduler and will integrate downlinked changes and uplink a revised short term plan.

o A ground scheduler, shown in Figure 1 as the planning function, is in the Platform Support Center. This scheduler is the most capable of the three and generates and maintains the initial schedule, and furnishes the short term plan to the PMS.

Platforms will be out of contact with the ground more than the manned base. As a consequence, platform operations management functions, both ground and space, need to be more autonomous than those of the manned base to effectively and efficiently meet mission goals. Automated replanning and rescheduling, including on-board real-time schedule maintenance and schedule repair, are required to support autonomous operation of platform systems and instruments.

PLATFORM SCHEDULING

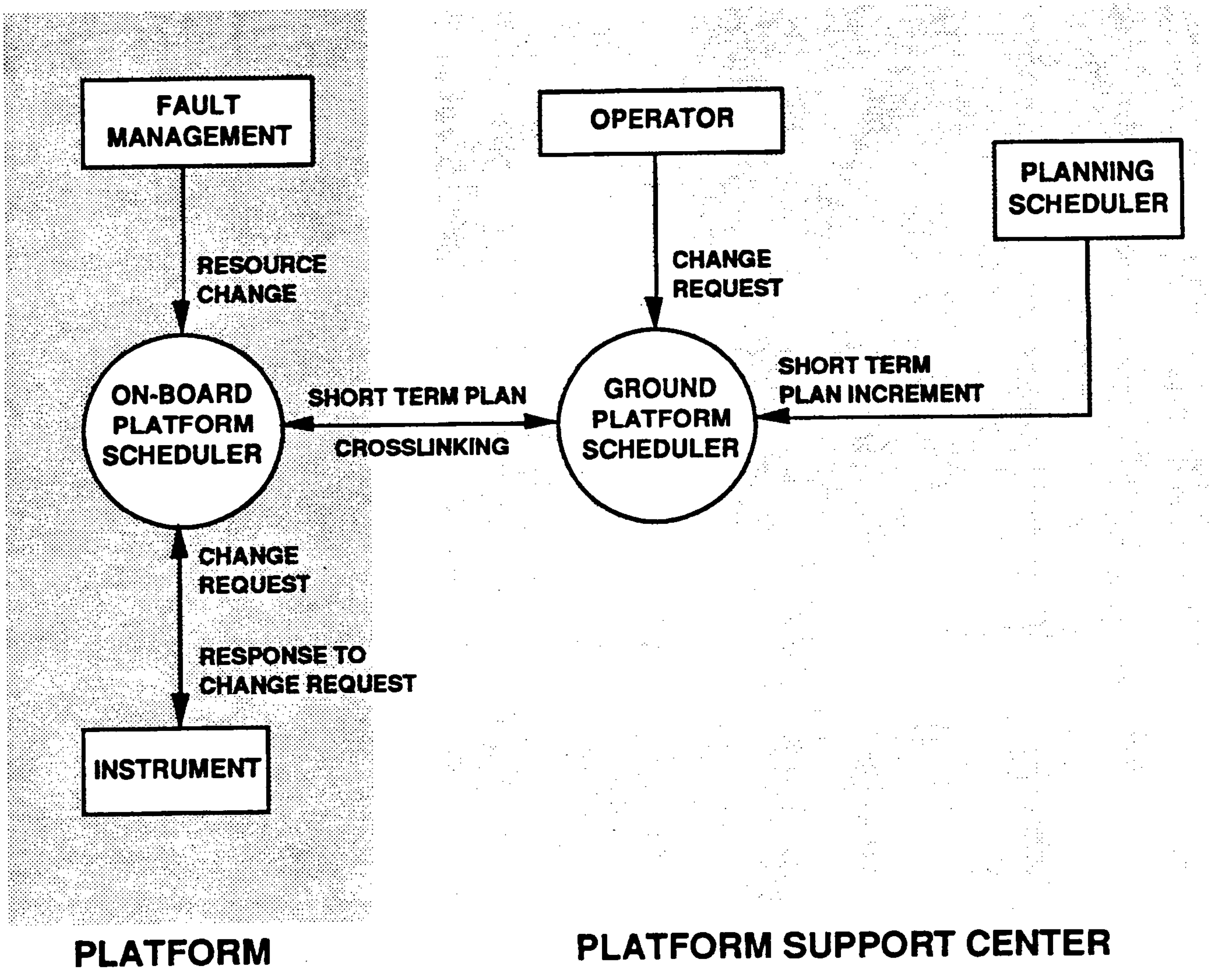


Figure 1

OBJECTIVES

Our FY88 study objectives were to analyze platform resource management, to generate functional requirements for platform scheduling and on-board plan management, and to develop promising alternatives for automation. We recommended both a specific alternative and a phased approach to automated platform resource scheduling. Our recommended alternative was based upon use of exactly the same scheduling engine in both ground and space components of the platform management system. Our phased approach recommendation was based on evolutionary development of the platform. The results of this study were published (References 2 and 3) and distributed in early 1989.

Our FY89 work focused upon implementation of our recommendation for platform resource scheduling in a manner that follows the phased approach and permits the scheduler to evolve over the life of the platform. We generated requirements specifications and designed a prototype platform management system scheduler. We also built a rapid prototype of this scheduler to explore some of the questions raised during the requirements and design work.

ORGANIZATION OF THE PAPER

The first half of this paper provides our rationale for the use of exactly the same scheduling engine for both components of the platform management system and our recommendation for evolutionary development. We begin with a definition of platform scheduling. Next, we introduce the twin problems of schedule maintenance and scheduler coordination. Having established the necessary foundation, we provide our rationale and recommendation.

The second half of this paper discusses our prototype platform management system scheduler. We describe the requirements for this platform scheduler, for on-board processing, and for ground processing. Next, we provide the requirements for crosslinking, a concept that we feel is essential to scheduler coordination. Following a brief description of our rapid prototype, we discuss our conclusions and one particularly subtle open issue under the heading of hooks and scars.

OBJECTIVES

FY88

- o Analyze platform resource management
- o Generate functional requirements
  - platform scheduling
  - on-board plan management
- o Develop automation alternatives
- o Recommend specific alternative/approach

FY89

- o Implement platform scheduler prototype
  - Generate requirements
  - Provide hooks and scars
- o Follow recommended phase approach

PLATFORM SCHEDULING

We define the platform schedule and both ground and on-board segments of this schedule as a set of envelopes arranged on a timeline. An "envelope", or "operations envelope", is a request for a set of resources to be allocated to instrument or platform for some period of time. Operations envelopes do not include commands to conduct the activity. A "resource" is either a measurable quantity or an environment in which to perform an activity that is provided by the platform to an instrument, e.g., an environmental right.

A schedule or short term plan is said to bear "conflicts" when either resources are oversubscribed or an environment is provided to one instrument that is not compatible with the desired environment of another instrument. In the case of the short term plan, conflicts may arise from three sources: instrument, end user, and platform. An example of an end user induced conflict is a request for more of a resource than is currently available, perhaps generated in response to a target of opportunity or other real-time event. A platform induced conflict results from unanticipated reduction in a resource.

Schedule Maintenance

We now define the maintenance problem for a platform scheduler: given a schedule, identify a segment of the schedule that contains conflicts and resolve those conflicts without affecting envelopes outside of the identified segment.

This task differs from that of a "planning" scheduler which generates the initial schedule. For comparison, we provide our definition of the schedule generation problem: given a set of requests, investigate different possible schedules in a search for a schedule that maximizes some figure of merit, e.g., number of requests scheduled.

Scheduler Coordination

We must also consider how the ground scheduler and the on-board scheduler will cooperate. The question of a scheme for cooperation arises because the on-board scheduler and the ground scheduler both have access to a copy of the short term plan and both receive requests to change it. This dual access poses the risk that both schedulers will alter their copies of the on-board plan at the same time. One new plan might not be compatible with the other.

PLATFORM SCHEDULING DEFINITIONS

OPERATIONS ENVELOPE

Request for a set of resources to be allocated to an instrument for some period of time

SCHEDULE / SHORT TERM PLAN

Set of envelopes arranged on a timeline

INITIAL SCHEDULE GENERATION

Given a set of requests, search for a schedule that maximizes some figure of merit

CONFLICT

A resource is oversubscribed or a an environment provided for one instrument is not compatible with the environment desired by another instrument

SCHEDULE MAINTENANCE

Given a schedule, identify a segment that contains conflicts and resolve without affecting envelopes outside the identified segment

SCHEDULER COORDINATION

Given two copies of a schedule, keep the copies compatible in the face of asynchronous and independent requests to change the schedule

We considered three possible ways to carve up the scheduling labor:

- o Concurrent Scheduling

The ground scheduler alters its copy of the short term plan when it receives a request. This is driven by a perceived need to be able to immediately tell a user who makes a change request whether or not the request can be scheduled. In the case of changes to the on-board portion of the plan, the ground scheduler incorporates the changes into its version of the plan.

- o Local Scheduling

The on-board scheduler schedules all of the requests that affect the on-board portion of the short term plan, and the ground scheduler handles all requests that affect the rest of the short term plan. This scheme prevents the system from being able to immediately tell users the status of their requests to change the on-board portion of the short term plan.

- o Pseudo-scheduling

The ground scheduler assists the on-board scheduler in making changes to the short term plan. When the ground scheduler receives a request that falls within the on-board span of the short term plan, it looks at its copy and determines how it would adjust the plan to accommodate the request. The ground scheduler does this without changing its copy of the short term plan. It creates a working copy. When the ground scheduler determines that it could satisfy the request, it saves the sequence of actions used along with the original request. If the ground scheduler is again asked to modify the short term plan, it repeats the procedure, but uses the working copy.

At the next contact, the on-board scheduler downlinks the master short term plan, and receives requests and sequences of actions from the ground scheduler. The ground scheduler then discards the working copy, and begins anew with the current on-board short term plan. When the on-board scheduler receives the request, it first tries the same sequence of actions taken by the ground scheduler. If it can do this without having a conflict occur, the request is scheduled in the way that the ground scheduler determined. If it cannot, then the on-board scheduler decides how to schedule the request on its own.

DIVISION OF SCHEDULING LABOR

CONCURRENT SCHEDULING

Ground Scheduler

- Alters its copy of the short term plan
- Provides user with immediate feedback

On-board Scheduler

- Provides on-board changes to ground
- Receives updated, altered plan from ground

LOCAL SCHEDULING

Ground Scheduler

- Alters only short term plan not yet uplinked
- Uplinks requests to change on-board plan

On-board Scheduler

- Alters only on-board portion of short term plan
- Downlinks requests to change remaining plan

PSEUDO-SCHEDULING

Ground Scheduler

- Assists on-board scheduler

On-board Scheduler

- Mimics ground scheduler's actions

PHASED APPROACH

We developed a conceptual model for implementation of the platform scheduler and for automation of platform scheduling. Our model is based upon an assumption that the platform itself will evolve over time. Our conceptual model provides for three stages of development over the life of the platforms. We do not presume to establish dates for each stage in the lifetime of the platform but simply name the stages of development: baseline, midterm, and final. These stages of development are shown in Figure 2 and discussed below.

o Baseline

Initially, we see both on-board and ground platform schedulers as simple schedule managers. Either local or concurrent scheduling may be followed. Given the need to be able to immediately tell a user who makes a change request whether or not the request may be scheduled, we assume that concurrent scheduling will be followed. The ground scheduler maintains the master copy of the short term plan and uplinks replacement for the on-board plan after first incorporating any on-board changes (simple safing actions) since the last contact.

o Midterm

At this stage, we see the on-board scheduler as a more sophisticated schedule manager with limited automated scheduling capability (enhanced safing) while ground scheduling is automated, but not yet autonomous. Pseudo-scheduling is followed with the ground scheduler uplinking both change requests and the sequences of actions that will schedule these requests provided the segment of the on-board plan affected has not changed since the last contact.

o Final

In the final stage, we see platform scheduling as both automated and autonomous. The platform scheduler takes the entire short term plan into account in resolving conflicts rather than dealing with limited segments. The platform scheduling requirement for scheduler coordination is satisfied by providing exactly the same scheduling engine in space and ground applications.

PLATFORM SCHEDULER DEVELOPMENT

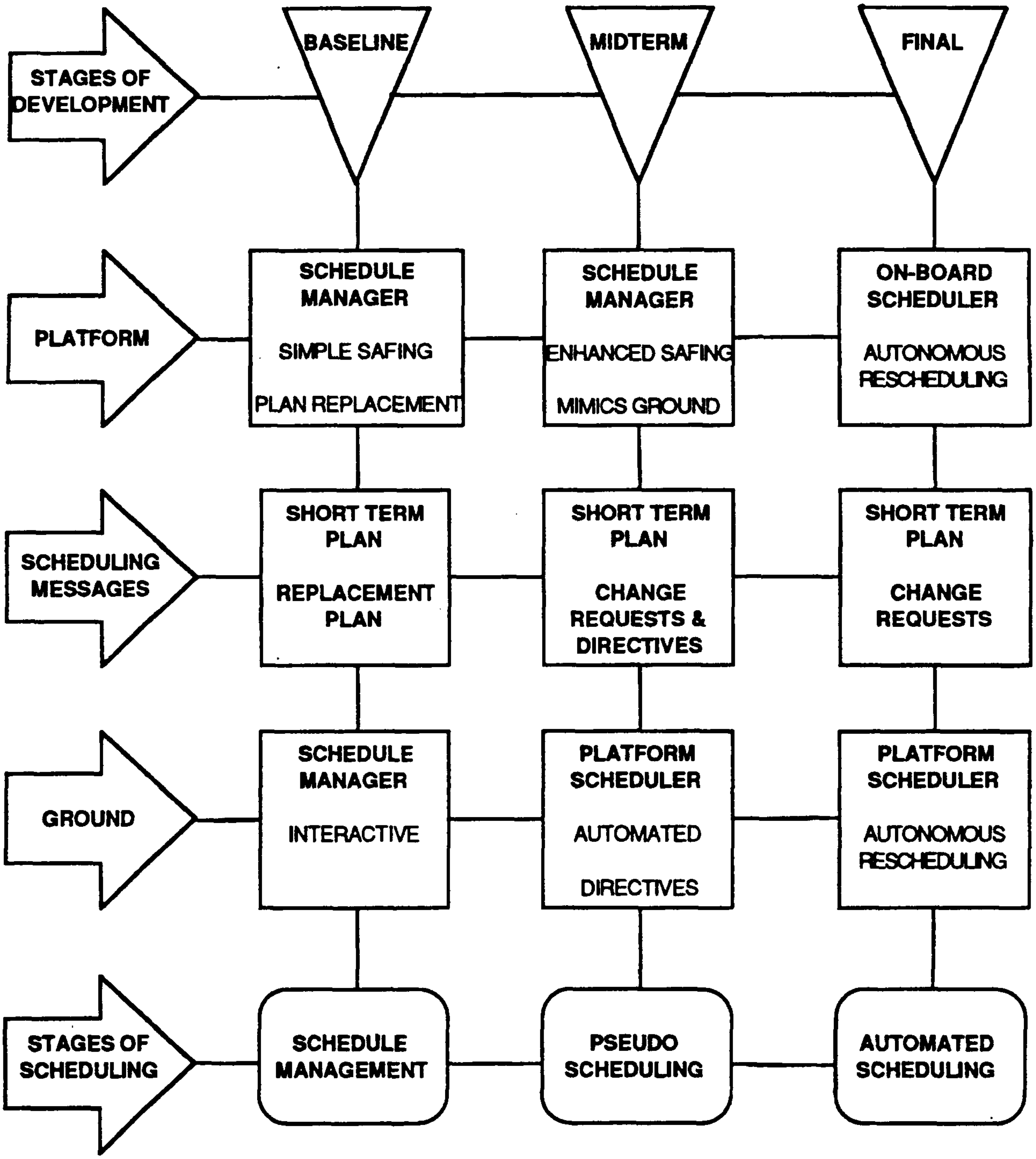


Figure 2

PROTOTYPE

As we have discussed, two platform schedulers are altering two schedules, with one schedule a subset of the other. The system must act in such a way that the ground and space components both agree on the on-board schedule immediately after each contact.

Prototype Operations Concept

Principal investigators submit requests for resources to the planning scheduler in the Platform Support Center. The planning scheduler generates the initial schedule and maintains the schedule through the start of the short term plan furnished to the platform schedulers. The planning scheduler forwards all requests that have a start time within the span of the short term plan.

Each request has a priority that the platform schedulers use to adjudicate conflicts. If two requests have the same priority and are in contention for the same resources, then we use the order in which the requests are received by the scheduler to determine a unique effective priority. A high priority, late arriving request can cause an existing but low priority request to be removed from the schedule.

A "smart" instrument may submit a change request to the on-board scheduler. This scheduler processes the request if it has a start time that falls within the current span of the on-board short term plan. If the request has a later start time, the on-board scheduler defers it to the ground scheduler at the next contact.

When fault management detects a change in platform resource capacities, it provides the on-board scheduler with the new resource availabilities. At the next contact with the ground, the on-board component "crosslinks" the schedules so that the space and ground applications have identical copies of the on-board short term plan and identical knowledge of the resource availabilities.

PROTOTYPE OPERATIONS CONCEPT

PLANNING SCHEDULER

- o Generates and maintains initial schedule
- o Furnishes short term plan to platform schedulers
- o Passes user change requests within span of plan

GROUND SCHEDULER

- o Processes all change requests within span of plan
- o Uses request priority to adjudicate conflicts
- o Crosslinks schedules and resource requests

ON-BOARD SCHEDULER

- o Processes only change requests within span of on-board plan
- o Uses request priority to adjudicate conflicts
- o Knows present platform resource availabilities
- o Crosslinks schedules and resource availabilities

BASELINE REQUIREMENTS

The operations concept discussed above allows many different sets of requirements, especially in connection with crosslinking. We used prototyping to identify one set of requirements that will allow this high-level operations concept. The requirements provided here are not the only requirements that will enable this operations concept.

Requirements on the Scheduling Engine

Our scheduling engine is a simple priority scheduler that allocates resources to requests depending upon resource availability and the priority of the request. For baseline capability, the scheduler needs to process only very simple kinds of requests. Each request has a specific start-time and duration, and includes a specification of all resources needed to accomplish some activity and the required environment conditions.

We assume that the baseline scheduler should allow the expression of some scheduling constraints in connection with the placement of a request on the timeline relative to other requests. However, these constraints have not yet been defined and our rapid prototype does not presently allow such scheduling directions.

With these simple requests, the scheduling engine satisfies three baseline requirements:

- o Do not schedule a request if that will oversubscribe resources.
- o Do not schedule a lower priority request if a higher priority request can be scheduled.
- o Maintain the schedule so that as many requests as possible are scheduled at all times.

BASELINE REQUIREMENTS

SCHEDULING REQUESTS

- Priority
- Start time
- Duration
- Resources
- Constraints
- Environmental Conditions

SCHEDULING ENGINE

- Do not schedule a request if that will oversubscribe resources
- Do not schedule a lower priority request if a higher priority request can be scheduled
- Maintain the schedule so that as many requests as possible are scheduled at all times

On-board Processing Requirements

The baseline on-board scheduler only adds or defers change requests from instruments. The on-board scheduler processes all requests that fall within the span of the on-board short term plan as well as those that fall outside the time span of the on-board plan by less than the period between regularly scheduled contacts. Those change requests with start times within one contact period of the end of the on-board short term plan would otherwise have to be downlinked, processed, and uplinked during the crosslink process, which is not necessarily going to be feasible.

The on-board scheduler must alter the priorities of requests dynamically if, as in our prototype, a simple priority scheduler is to be used. It is the simplest way to prevent the ground scheduler from removing requests scheduled on-board. It ensures that the ground and space components have the same version of the on-board schedule immediately after each contact.

No request is submitted to our prototype, whether acting as the on-board scheduler or the ground scheduler, with a priority greater than 4. We increase the priority of any request scheduled on-board so that it is in a range from 5-9. Further, a request that is active (start time less than current time) is given the highest priority of 10. This scheme, while not the only possible alternative, does guarantee two necessary characteristics of the schedule maintained by our priority-based scheduler prototype:

- o Since active envelopes are given the highest priority, the scheduler will remove active envelopes from the schedule in response to a degradation in resources only as a last resort.

- o When the schedules (on-board and ground versions of the short term plan) are merged on the ground, all requests scheduled on-board will be scheduled as well by the ground scheduler.

BASELINE REQUIREMENTS

ON-BOARD PROCESSING

- o Add or defer change requests from instruments
- o Process requests within the time span of the on-board plan
- o Process requests within one contact period beyond the time span of the present on-board plan
- o Defer all requests beyond the present span plus the time between ground contacts (nominally one orbit)
- o Alter the priorities of the scheduled requests
- o Remove active envelopes from the schedule only as a last resort

Ground Processing Requirements

The baseline ground scheduler both adds and deletes requests. The ground scheduler processes all requests that fall within the span of the short term plan. It also merges the on-board versions of the short term plan into the ground short term plan during crosslink.

The requests that the ground scheduler processes (ending with a status of either scheduled or not) and that fall within the span of the on-board short term plan are uplinked at the next contact period. The on-board short term plan time span is extended by the time between contacts at the start of each contact, just prior to crosslink.

Crosslinking Requirements

The crosslink process is the sequence of steps that the on-board and the ground scheduling systems must accomplish to ensure that the on-board short term plan and the corresponding portion of the ground short term plan are exactly the same immediately after each contact.

Our scheduler prototype implements the crosslink process in three steps:

o The crosslink is made at a regularly scheduled contact time (perhaps once each orbit) and both schedulers increase the time span of the on-board short term plan by one contact period.

o The ground scheduler uplinks all requests with a start-time that falls within the (updated) span of the on-board short term plan. The on-board scheduler adds them to the schedule one-by-one and screens for conflicts after each addition. At the completion of this process, the platform has an executable on-board short term plan.

o As the final step, the on-board scheduler sends the resource availabilities, the on-board short term plan, and all deferred and unscheduled requests to the ground. The ground scheduler merges the present on-board plan with the rest of the short term plan, screens the new schedule against the current resource availabilities, and processes all deferred requests.

BASELINE REQUIREMENTS

GROUND PROCESSING

- o Add or delete change requests
- o Process requests within the time span of the short term plan
- o Uplink requests within the time span of the on-board schedule at the next contact

CROSSLINKING

- o Crosslink at a regularly scheduled contact time
- o Increase time span of the on-board plan by the interval between contacts prior to crosslink
- o Uplink all requests with a start time that falls within this time span
- o Downlink on-board plan, deferred requests, unscheduled requests and resource availabilities

RAPID PROTOTYPE DESCRIPTION

We built a rapid prototype of the platform scheduler to explore some of the questions raised during the requirements and design work. This rapid prototype is designed to be both the on-board scheduler and the ground scheduler. As the on-board scheduler, the prototype acts in exactly the same way as the ground scheduler except that it dynamically adjusts the priorities of requests it can schedule and in execution. We use this dynamic adjustment of priorities to prevent requests that are scheduled on-board from being unscheduled on the ground and to guarantee, that in instances of resource degradation, the on-board scheduler will not remove active requests except as a last resort.

Our rapid prototype implements both request management and conflict recognition and resolution functions. Request management first determines whether to process (add, delete, replace) a request or to defer a request. A request is deferred if it falls outside the span of the current short term plan. After all requests have been processed, the conflict recognition and resolution function is called to ensure a conflict free plan. If a conflict is found, this function resolves it by unscheduling all requests at that time and then attempting to add them back to the schedule in priority order.

Unscheduling differs from deleting a request. The rapid prototype will unschedule lower priority requests to accommodate a higher priority request. However, our prototype does not remove the unscheduled requests from the schedule. It only changes the status of these requests. We retain unscheduled requests since subsequent changes may allow these requests to be rescheduled, e.g., higher priority requests may be unscheduled or deleted.

The rapid prototype is menu-driven as shown in Figure 3. Our implementation allows the user to crosslink at any time. When crosslink is selected, the rapid prototype sequences through the crosslink steps waiting only for the user to grant permission to proceed. This manual capability enables us to easily demonstrate crosslinking. A fully automated capability will be needed to support emergency crosslink.

RAPID PROTOTYPE MENU SYSTEM

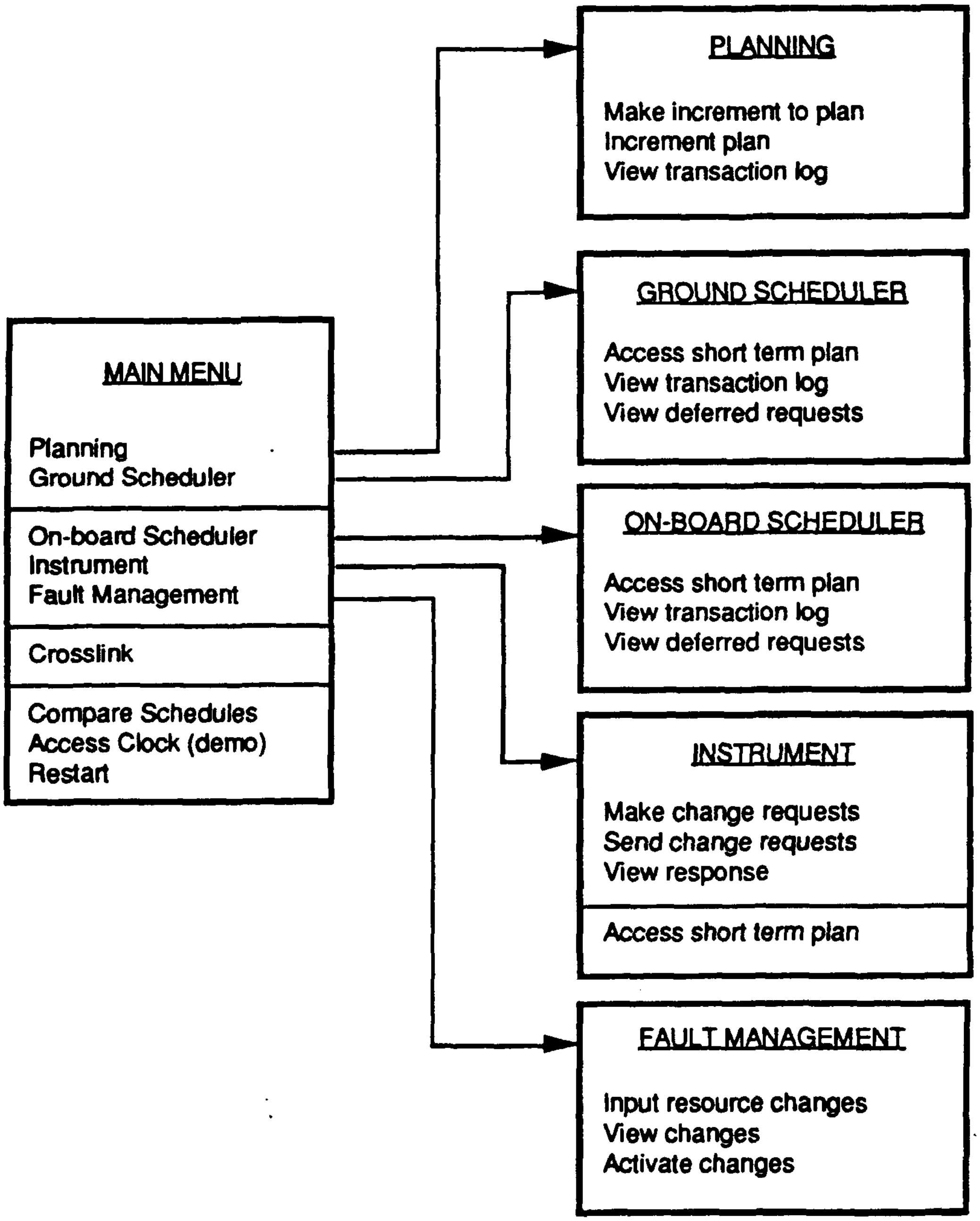


Figure 3

HOOKS AND SCARS

The prototype platform scheduler work and our rapid prototype were guided by the requirements generated in our FY88 study. These requirements, and the accompanying methodology for evolution, did not use hooks as the mechanism for evolving the capabilities of the scheduler. We relied on module replacement.

Module replacement is a reasonable strategy for evolution, but requires sufficiently powerful data structures at the beginning of the life cycle. These data structures should, even in the baseline, provide all of the information to the platform scheduler that it will need in order to automatically, autonomously reschedule.

The envisioned data structures will express all possible ways that the platform scheduler can satisfy the need for resources in support of an activity. By the final stage in development, if any activity must be removed to make room for a higher priority request, the scheduler will look at the request for this lower priority activity to see how it can be rescheduled.

As emphasized, unscheduling differs from deleting a request in our rapid prototype. The prototype will unschedule requests in order to accommodate a higher priority request. Unscheduling only changes the status of these requests. Request management tries to add these requests back to the schedule when either resource availabilities change or a higher priority request is deleted from the schedule.

One complication in making changes to the schedule may not be immediately obvious. A request for one resource can imply a request for another resource. Power may be an implied resource. The actual ceiling for power varies not only because of the need for power to run the platform and operate instruments, but also for requested resources that only imply the use of power, such as a tape recorder. Further work is needed on this issue.

HOOKS AND SCARS

MODULE REPLACEMENT

- o Provides a reasonable strategy for evolution
- o Requires sufficiently powerful data structures at baseline

ENVISIONED DATA STRUCTURES

- o Express all possible ways to satisfy a request
- o Reduce number of unscheduled requests
- o Improve utilization of platform resources

UNSCHEDULING REQUESTS

- o Differs from deleting requests
    - Status flag is changed
    - Request may still be accessed
- o May reschedule previously unscheduled requests
    - Higher priority request is itself deleted or unscheduled
    - Resource availabilities change

IMPLIED RESOURCES

- o Implied in a request for another resource

ACKNOWLEDGEMENTS

The work described in this paper involves a large number of people. This work could not have been performed without the support of Gregg Swietek, Strategic Plans and Programs Division, Office of Space Station. Specific thanks are due to Don Rosenthal, Ames Research Center, whose presentation of lessons learned at the Boulder workshop (Reference 4) helped to guide this work. Finally, credit for specific concepts and for implementation of the rapid prototype of the schedule belongs to James Retter, System Sciences Division, Computer Sciences Corporation.

REFERENCES

1. Goddard Space Flight Center, "Platform Management System (PMS) Definition Document", Revision 3, September 1987.

2. Computer Sciences Corporation, "Methodology for Platform Management System (PMS) Scheduling Automation", CSC/TM-88/6144, March 1989.

3. Computer Sciences Corporation, "Platform Management System (PMS) Scheduling Requirements Study", CSC/TM-88/6036, March 1989.

4. Rosenthal, Donald, "Lessons Learned from HST Scheduling", Workshop on Operations Planning and Scheduling Systems for the Space Station Era, University of Colorado at Boulder, Boulder, Colorado, August 1987.

5. NASA, "Space Station Program Definition and Requirements Document", Section 4, Part 3: Space Operations Requirements, Revision B, SSP 30000, Space Station Program Office, Reston, Virginia, October 1988.

CONCEPTS IN DISTRIBUTED SCHEDULING AND CONTROL

Elaine R. Hansen
Thomas P. Sparn
Laboratory for Atmospheric and Space Physics
University of Colorado at Boulder
Boulder, Colorado 80309

Larry G Hull
Code 522
Goddard Space Flight Center
Greenbelt, Maryland 20771

ABSTRACT

To support instrument and experiment operations effectively in the Space Station era, planning, scheduling and control must allow for:

- o interactive, realtime, remote operations;
- o responsive scheduling and rescheduling;
- o support of the full range of distributed science, application and commercial users;
- o interaction and cooperation among distributed users; and
- o efficient use of on-board, communications, and ground-based resources.

We suggest conceptual and managerial approaches that address these needs.

Specifically, we describe an approach to distributed planning, scheduling and control functions that is based on resources and on a distributed knowledge hierarchy. We describe these functions as components of an integrated management system. We discuss automated scheduling assistants and integration of planning and scheduling functions with realtime operations control.

The suggested approach, taken from a users' point-of-view, has resulted in the Science User Resource Planning and Scheduling System (SURPASS). In this paper, we describe the major components of SURPASS and discuss the features of this innovative prototype. Further ideas concerning instrument planning, scheduling, and control may be found in the Space Station Instrument Control System Study (Reference 1).

INTRODUCTION

The space station era will open new and unique opportunities by making the environment of space accessible to a large community of scientists as a scientific laboratory. Some aspects of this space laboratory include low gravity environment, low pressure, no atmospheric attenuation, the ability to make global observations and complete celestial viewing. A noteworthy aspect of this space laboratory is the large separation between the scientists on the ground and their in-space experiments. To assist the scientist in interacting with their far off laboratory, an approach called "telescience" will be used. Telescience takes advantage of telecommunication services to allow scientists to remain at their home institutions where they can fully interact with their in-space experiments; where they can fully participate in planning, scheduling, controlling, evaluating and refining these experiments; and where they can work along side their research colleagues and students.

Supporting technologies and several new concepts in distributed scheduling and control need to be developed and demonstrated in order to fully support this distributed laboratory environment. These concepts are shown in the facing bullet chart.

ORGANIZATION OF THE PAPER

We begin by describing previously proposed approaches to planning and scheduling by distributed users. Next, we discuss the work being performed at the University of Colorado in distributing instrument scheduling and control. This discussion focuses on the Science User Resource Planning and Scheduling System (SURPASS), a knowledge based prototype supported by Goddard Space Flight Center and the Strategic Plans and Programs Division of the Office of Space Station. After a description of SURPASS, we present conclusions based upon our initial work.

CONCEPTS SUPPORTING THE DISTRIBUTED LABORATORY ENVIRONMENT

o Provide for interaction between user scientists and their remote experiments.

o Support scientists and commercial users involved in a wide range of applications.

o Enable scheduling and rescheduling of experiment activities by distributed science users.

o Make efficient use of on-board resources and promote optimal scheduling of these resources.

o Allow the user scientist to quickly reschedule activities to react to science opportunities or problems.

o Allow the science and applications users to work at their home institutions.

PREVIOUS APPROACHES

In the past, a number of systems have been proposed that address planning and scheduling by distributed users. These systems were largely based on a centralized planning and scheduling hub with remote scientists placing requests on a single large global database via communications networks. Figure 1 conceptually illustrates the centralized scheduling approach.

Typically, in these proposed centralized scheduling approaches, users make requests for an experiment to be initiated at a specific time to make a specific observation. The resultant schedule to support these multiple user requests is only assembled after all these requests are received at the central hub. These requests are usually required several days to several months before the schedule is to be generated and executed, in order to allow sufficient time for constraint checking and activity scheduling. This large lead time makes it difficult, often impossible, to reschedule experiments on short notice. This inflexibility prevents experimenters from refining their experiments based on progress or responding to unexpected events. Inflexibility is both an attribute associated with centralized planning and scheduling and a characteristic of the centralized scheduling systems we have examined.

In direct contrast to the inflexibility of the planning and scheduling support, scientific instrumentation has been progressing to allow for more flexibility. Smart instruments, with embedded microprocessors, have or are expected to soon become the standard. The embedded microprocessors extend instruments' capabilities and allow an experiment to adapt to experiment findings or external conditions. These microprocessors also protect the instrumentation by automatically responding to anomalies or out-of-tolerance conditions. These more flexible and responsive instruments need planning, scheduling and control systems that support their evolving needs and capabilities.

CENTRALIZED SCHEDULING APPROACH

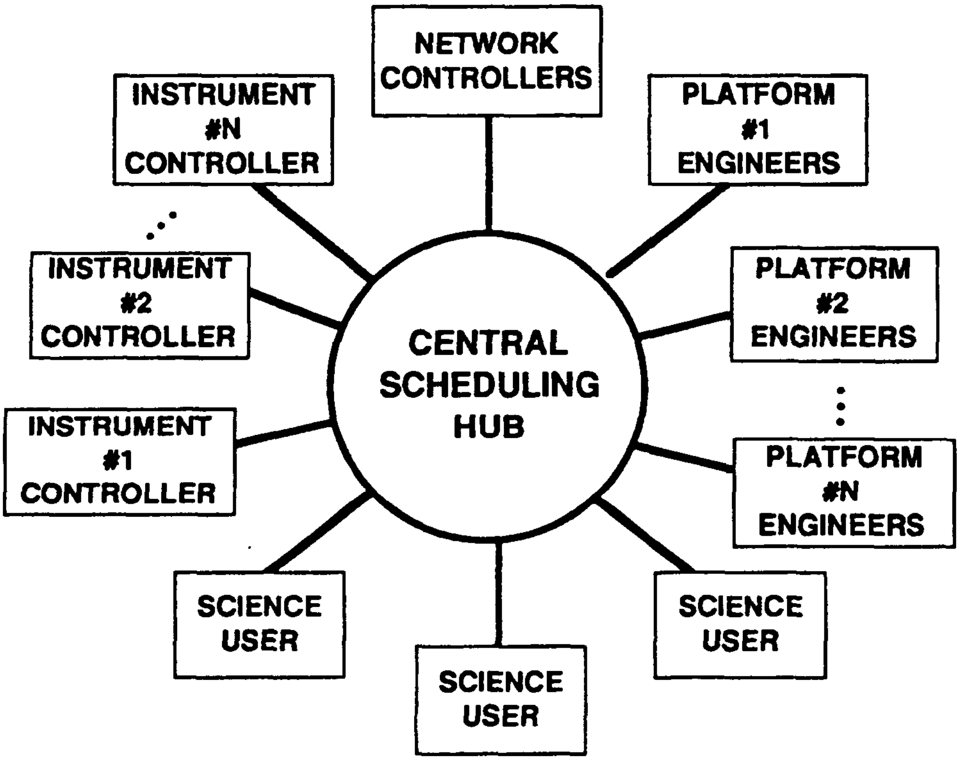


Figure 1

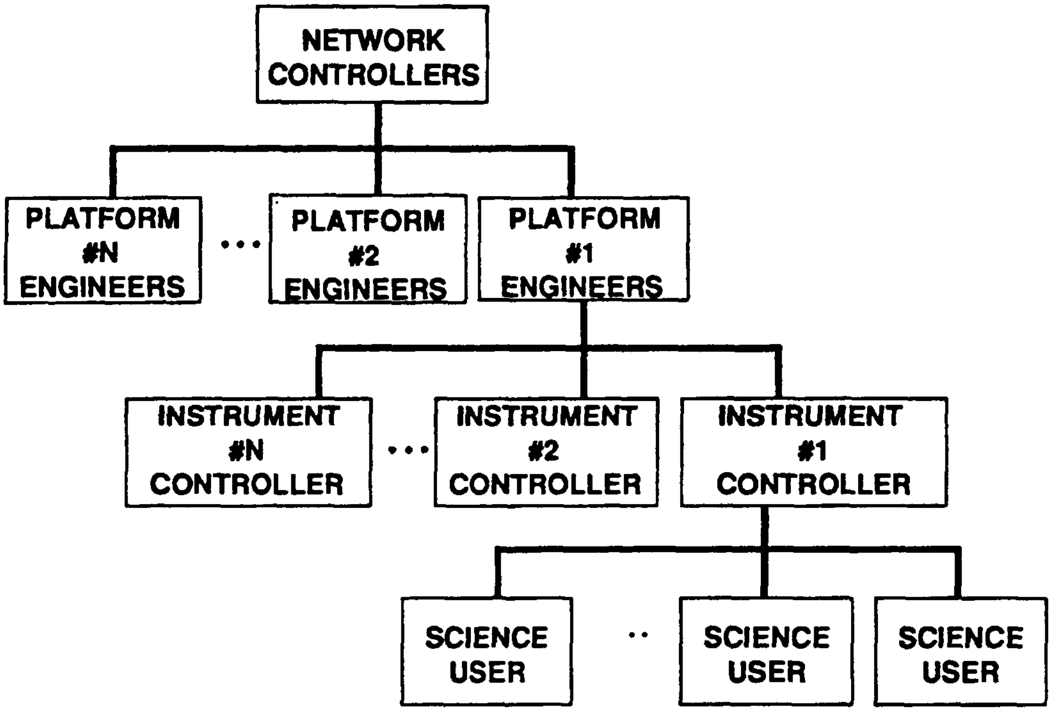
DISTRIBUTED SCHEDULING AND CONTROL

Work at the University of Colorado has centered around distributing the scheduling and control activity among many geographically distributed nodes. In the evolving concept, each node is responsible for scheduling those activities with which it is most concerned. Individual science users are each responsible for developing a schedule of their own science experiments. The engineers at a platform support facility are responsible for scheduling platform activities such as attitude maneuvers, battery operations, and tape recorder maintenance. Similarly, controllers of communications services are responsible for scheduling these resources.

The geographically distributed nodes are arranged in such a way that they form a hierarchy that is leveled according to the physical systems which accomplish the goal. An illustration of this type of leveling is shown in Figure 2. In this hierarchical representation, the controllers of communications and tracking services, who are responsible for supporting a range of space missions, are at the top of the hierarchy. At the next lower hierarchical level are the engineers responsible for the platforms. Below them are the scientists and engineers responsible for the health and performance of the science instruments. At the lowest level of this conceptual hierarchy are the research scientists, who have the responsibility for the experiment program. At each level, and at each node in this hierarchy, local schedule optimization is accomplished using the knowledge present at that level and the predicted availability of the resources that will support the scheduled activities. In this distributed arrangement, rescheduling can be accomplished quickly when a change is requested that does not require rescheduling at the global level.

To support communications among the distributed scheduling nodes in this hierarchy, a common "language" is needed. This language needs to be able to flexibly and comprehensively communicate scheduling opportunities and user requests. The University of Colorado and Goddard Space Flight Center have developed the Flexible Envelope Request Notation (FERN), a prototype scheduling applications interface language (Reference 2) that addresses this need.

DISTRIBUTED SCHEDULING APPROACH



SCHEDULING HIERARCHY

Figure 2

PROTOTYPE TOOLS

Over several years, the researchers at the University of Colorado's Laboratory for Atmospheric and Space Physics (LASP) have developed a set of prototype tools to help demonstrate how physically separated science teams will participate in planning and scheduling activities. The first phase of this work was part of the Telescience Implications on Ground Systems (TIGS), a Code T-funded study performed with the Data Systems Technology Division at Goddard Space Flight Center. This study was completed in early 1988. It was primarily concerned with the problems of communicating resource needs and opportunities among distributed nodes.

SURPASS

The second phase, which has just started, is focused on the design and development of a knowledge based Science User Resource Planning and Scheduling System, (SURPASS). This prototype will be integrated into the Scheduling Concepts, Architectures, and Networks testbed at Goddard Space Flight Center. Use of SURPASS in this testbed will demonstrate the distributed planning, scheduling, communications, and operations management concepts. Figure 3 shows the three components of SURPASS: an expert system scheduling aid, Science User Resource Expert (SURE), an adaptable user interface that can easily be tailored to the science user's picture of the scheduling activity, and a Planning and Scheduling System manager (PASS).

SURPASS is designed to enable the remote user scientist to fit into the planning and scheduling hierarchy while maintaining the fidelity of planning and optimizing his or her experiment activities based on local scientific goals and considerations. The user interface allows the user to schedule within the appropriate scientific context and is adaptable to ensure that the SURPASS interface is consistent with other data system interfaces used by the scientist. SURE aids the user scheduling experiment activities to take optimal advantage of the available resources and still fit within resource constraints. PASS buffers complex data structures and handles communications, transactions, and interfaces.

SCIENCE USER RESOURCE PLANNING AND SCHEDULING SYSTEM (SURPASS)

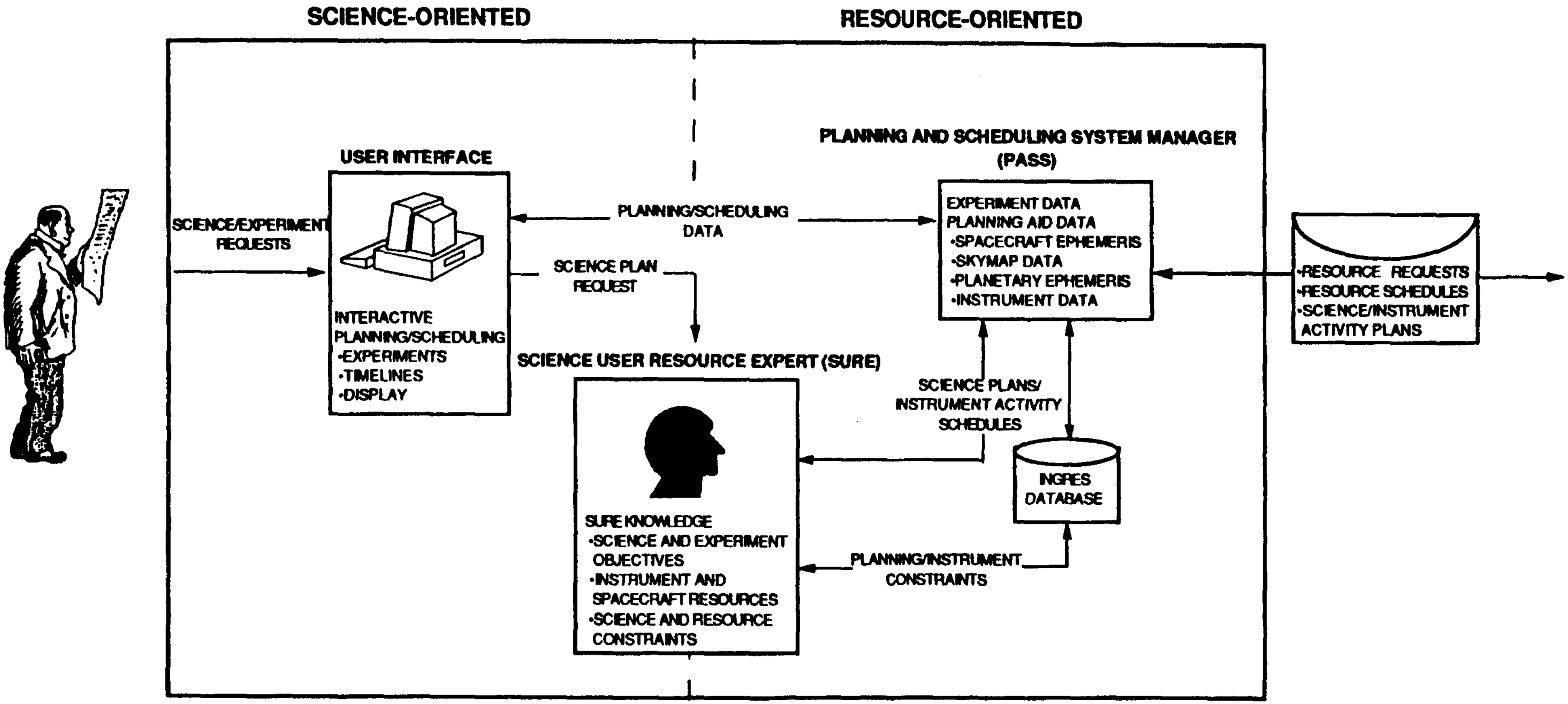


Figure 3.

Starting from an initial allocation of platform/spacecraft resources, the expert planner, SURE, builds an experiment plan based upon broad science goals and detailed relations of instrument activities and resource needs. Figure 4 provides examples of the experiment activity and resource usage which SURE attempts to maximize. SURE does not, however, have the more extensive knowledge needed to change science objectives. This is supplied by the science user.

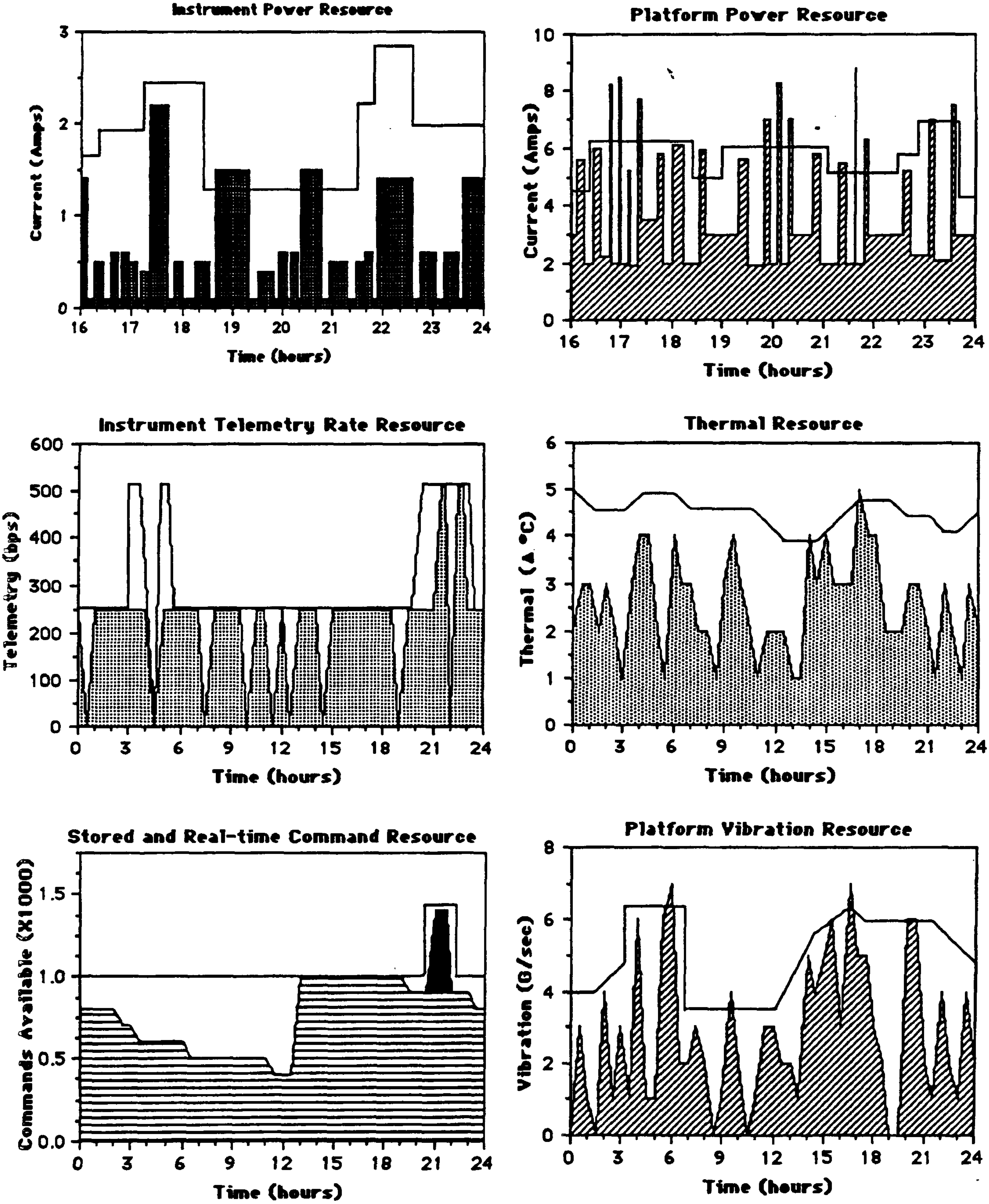
The current prototype schedules experiment activities for the SOLSTICE instrument to be flown on the UARS mission. In this prototype, SURE generates an initial plan with a feasible acquisition sequence for a set of candidate stars selected according to the rules provided. To illustrate its performance, the time to generate a one day plan by exhaustive search required some 40 hours. Use of the SURE system with heuristics reduced the time to schedule a 24 hour day to only 15 minutes.

The output from the expert planner is displayed by SURPASS in terms of science coverage rather than the individual resources needed. The user scientist may then adjust the science plan by adding or modifying experimental activities. The SURE system calculates resource changes, checks constraints, and attempts to fit the activity into the timeline. SURE notifies the user if the activity plan cannot be inserted due to resource or constraint conflicts.

As an aid to conflict resolution, several windows providing additional information may be dynamically requested by the user. These windows inform the user of what constraint is being violated or what resource is insufficient and should be re-negotiated.

SURPASS generates schedule requests through the Planning and Scheduling System (PASS) manager. The PASS manager uses the scheduling applications interface language to communicate instrument activity requests to a platform resource scheduler in terms of resource envelopes. Complex data structures are hidden from the user by the PASS manager and maintained within the INGRES database management system. This software translates inputs into internal SURPASS data structures and internal data structures into outputs in the scheduling applications interface language.

SOLSTICE/SSPP Resource Usage 1992/033



Instrument Power Resource

Platform Power Resource

Instrument Telemetry Rate Resource

Thermal Resource

Stored and Real-time Command Resource

Platform Vibration Resource

Figure 4.

CONCLUSIONS

This research deals with the broad scope of the scheduling and control problem with its large and changing number of users, the large number of experiment plans that must be integrated each day, and the need for scientific flexibility and evolution. Some initial conclusions are summarized below:

- o Science is an exploratory activity. The scientific method, whether the investigation takes place in a small room or in a large distributed laboratory or in space, is an interactive process wherein the experimenter continually refines the investigation based on experiment findings and external changes.

- o Science users can plan and schedule instrument activities with respect to their own scientific goals. These goals can be translated into resource requests.

- o Expert systems and knowledge based tools are ideal for generating a science experiment schedule which satisfies science observing objectives, complies with rules and constraints, and remains within the available schedule of resources.

- o A distributed scheduling system with local scheduling at each node by interested, knowledgeable users is both possible and valuable.

- o Breaking down the scheduling problem into levels, and nodes within each level, the subset of the scheduling problem becomes tractable. This approach allows knowledgeable users to resolve the planning and scheduling issues locally and reschedule activities without affecting the scheduled activities of other nodes.

- o A common scheduling applications interface language is needed to be able to flexibly and comprehensively communicate scheduling opportunities and user requests among the many physically separate scheduling nodes.

CONCLUSIONS

- Science is an exploratory activity

- Science users can plan and schedule instrument activities with respect to their own scientific goals

- Expert systems and knowledge based tools are ideal for generating a science experiment schedule

- A distributed scheduling system is both feasible and desirable

- Subdivided, the scheduling problem becomes tractable

- A common scheduling applications interface language is necessary

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the support and guidance that we have received throughout this study from Gregg Swietek of the Strategic Plans and Programs Division, Office of Space Station.

This work has been implemented and demonstrated by a team of enthusiastic and quality professionals and students including Daniel Gablehouse, William Gregg, Ronald Oakley, and Nancy Thalman.

REFERENCES

1. Data Systems Technology Laboratory, "Space Station Instrument Control System Study, DSTL-89-013, November 1989.

2. Data Systems Technology Laboratory, "User's Guide For The Flexible Envelope Request Notation (FERN)", DSTL-89-015, September 1989.

3. Laboratory for Atmospheric and Space Physics, University of Colorado, Ford Aerospace Corporation, Goddard Space Flight Center, Data Systems Technology Division, "Interface Control Document for the Telescience Implications on Ground Systems (TIGS) Joint Effort", January 1988.

4. Hansen, Elaine, Sparn, Thomas, Davis, Randal, "Concepts for Planning and Scheduling in the Space Station Era", Laboratory for Atmospheric and Space Physics, University of Colorado, Boulder, Colorado, May 1988.

5. Hansen, Elaine, "Requirements for Space Station Telescience: Command, Control, and User Interface Technologies", Laboratory for Atmospheric and Space Physics, University of Colorado, Boulder, Colorado, August 1986.

6. Hansen, Elaine, Davis, Randy, Faber, Jack, Jouchoux, Alain, Ludwig, George, "Telescience Testbed Program Results", OIS 89-1, Laboratory for Atmospheric and Space Physics, University of Colorado, Boulder, Colorado, March 1989

7. NASA, "Space Station Program Definition and Requirements Document", Revision B, SSP 30000, Space Station Program Office, Reston, Virginia, January 1989.

8. "Space Station Information System (SSIS) Concept Document", Draft, February 1988.

9. "Space Station Information System (SSIS) Systems Requirements Document (SRD)", April 1988.

10. Hansen, Elaine, Ludwig, George, Davis, Randal, Jouchoux, Alain, "Information System Scenarios for Space Science and Application", Review, Laboratory for Atmospheric and Space Physics, University of Colorado, Boulder, Colorado, August 1989.

REPORT DOCUMENTATION PAGE

1. Report No.
NASA CP-10044

2. Government Accession No.

3. Recipient's Catalog No.

4. Title and Subtitle
Beyond the Baseline - Proceedings of the Space Station Freedom Evolution Symposium at League City, Texas, February 6, 7, & 8, 1990
Volume 2: Space Station Freedom Adv. Dev. Prog.; Part 1

5. Report Date
April 1990

6. Performing Organization Code
HQ MT

7. Author(s)
NASA and support contractor personnel funded under the Transition Definition line item - UPN 488

8. Performing Organization Report No.
S-606

9. Performing Organization Name and Address
Space Station Engineering
NASA Headquarters
Washington D.C. 20546
MC:MT

10. Work Unit No.

11. Contract or Grant No.

12. Sponsoring Agency Name and Address
National Aeronautics and Space Administration
Washington D.C. 20546

13. Type of Report and Period Covered
Conference publication

14. Sponsoring Agency Code

15 Supplementary Notes

16. Abstract

This report contains the individual presentations delivered at the Space Station Evolution Symposium in League City, Texas on February 6, 7, 8, 1990. Personnel responsible for Advanced Systems Studies and Advanced Development within the Space Station Freedom Program reported on the results of their work to date. Systems Studies presentations focused on identifying the baseline design provisions (hooks and scars) necessary to enable evolution of the facility to support changing space policy and anticipated user needs. Also emphasized were evolution configuration and operations concepts including on-orbit processing of space transfer vehicles. Advanced Development task managers discussed transitioning advanced technologies to the baseline program, including those near-term technologies which will enhance the safety and productivity of the crew and the reliability of station systems. Special emphasis was placed on applying advanced automation technology to ground and flight systems.

17. Key Words (Suggested by Author(s))
evolution; "hooks and scars"; advanced automation; knowledge-based systems; robotics

18. Distribution Statement
Unclassified - unlimited

Subject category: 18

19. Security Classification (of this report)	20. Security Classification (of this page)	21. No. of pages	22. Price
U	U	344	

For sale by the National Technical Information Service, Springfield, VA 22161-2171

JSC Form 1424 (Rev Jan 88) (Ethernet Jan 88)